UNIVERSIDAD ESAN

FACULTAD DE INGENIERIA

DESARROLLO DE VIDEOJUEGO PARA EL APRENDIZAJE DE VOCABULARIO DEL IDIOMA INGLÉS.

Tesis para optar el Título de Ingeniero de Tecnologías de Información y Sistemas

Fernando Leoncio Flórez Ponce

Asesor: Marks Calderón

Lima, abril, 2018

Esta Tesis denominada:

DESARROLLO DE VIDEOJUEGO PARA EL APRENDIZAJE DE VOCABULARIO DEL IDIOMA INGLÉS.

ha sido aprobada.

..……..……..……..……..……..
Javier Fernando Del Carpio Gallegos (Jurado Presidente)

..……..……..……..
Nicolás Francisco Figueroa Mosquera (Jurado)

……..……..……
Wilfredo Mamani Ticona (Jurado)

Universidad Esan

2018

ÍNDICE

INDICE DE FIGURAS

INDICE DE TABLAS:

Agradecimientos:

Este trabajo de tesis realizado para la Universidad ESAN como un esfuerzo conjunto en el cual varios factores participaron para completa resolución de dicho documento. Este trabajo me ayudado a aprovechar la competencia y experiencia de diversos actores que deseo agradecer a continuación.

A mis padres, por ser siempre pacientes y comprensivos conmigo. Más que para mí, esta tesis es para ellos en especial, ya que esperan que tenga un buen futuro por delante, sin importar los obstáculos que se presenten.

A mis amistades, por darme ánimos para seguir adelante y terminar este largo proyecto, dándome apoyos para que no pierda voluntad en su desarrollo.

A mi asesor Marks Calderón, el cual me hizo comprender que soy capaz de aprender conocimientos necesarios para el desarrollo del proyecto de tesis en el menor tiempo posible. Su experiencia y afabilidad fueron muy bienvenidos para el desarrollo de esta tesis.

Y, finalmente, al profesor Joseph Ballón, el cual me dio las bases de programación en C#, el cual me ha sido de utilidad a través de mis siguientes años fuera de la universidad.

RESUMEN:

El presente estudio consistió en desarrollar un videojuego con el propósito de afrontar el problema de la educación en relación a la falta de conocimiento del idioma inglés en temas de vocabulario. La efectividad de la aplicación de los videojuegos en la educación en otros países del primer mundo resultó en una mejoría del índice de aprendizaje de los alumnos involucrados.

El trabajo consistió en crear un videojuego con capacidad de actuar como una herramienta para los profesores de inglés para que puedan ser capaces de mejorar el nivel de atención y rendimiento de sus alumnos por medio de la aplicación de redes neuronales, con el propósito de que los alumnos de sector primaria y secundaria puedan asimilar mejor los conocimientos lingüísticos para su educación en general, debido a la necesidad actual del idioma para competir a nivel mundial.

El trabajo en sí consistió darle más contenido al videojuego en cuestión por medio de la aplicación de técnicas de ciencias de la computación actual para mejorar la experiencia del alumno para optimizar el aprendizaje, tomando en consideración los avances actuales tanto en la tecnología como en el entretenimiento.

Entre los resultados más importantes se encontraron que los beneficios de utilizar los videojuegos para la enseñanza incluyeron la facilidad de aprendizaje y una mejor asimilación de conocimientos. Se encontraron que los videojuegos son mejores herramientas debido a la facilidad de entender y experimentar, al no generar tanta resistencia como el aprendizaje común, al mostrarse una 80% de aceptación encontrada al probar la hipótesis.

INTRODUCCION:

El objetivo de la investigación busca justificar la necesidad de un videojuego educativo para mejorar el nivel de aprendizaje de inglés de los alumnos peruanos con rango de edad entre 5 y 11 años en relación al aprendizaje del vocabulario. De esta manera, se buscará mejor el índice de aprendizaje del idioma inglés y mejorar el nivel educativo del Perú.

Actualmente, el nivel de educación a nivel nacional se encuentra desbalanceado. Sectores rurales fuera de Lima siguen sin tener las mismas capacidades educativas, generando una falta de alumnado con capacidades necesarias para competir a nivel nacional y/o mundial. Aunque la situación ha mejorado considerablemente en comparación al nivel educativo antes del inicio del milenio, la educación sigue siendo subpar en comparación a países como Chile.

El desarrollo de este videojuego educativo se centraría a solucionar el problema educativo de los alumnos en sectores de nivel económico bajo (C hacia abajo), usando un buen diseño en general del videojuego planeado y de técnicas de conocimiento informático para brindar mayor contenido al alumno, con el propósito de mantenerlo entretenido y así, que el mismo se preste atento al conocimiento brindado.

Para el desarrollo del proyecto, se investigará las técnicas aplicadas por otras instituciones en relación al tema de aprendizaje mediante los videojuegos alrededor del mundo y buscar aplicarlas o mejorar las mismas, en relación al contexto peruano de educación para difundirlo vía Internet para las áreas con los menores niveles de educación, solucionando así el problema del desnivel educativo.

En el primer capítulo, se describe la problemática de investigación, determinando problemas y sus objetivos principales, y la justificación del estudio.

En el segundo capítulo, se observan los principales antecedentes de la investigación centrándose en el ámbito de educación por medio de videojuegos y los resultados de las mismas investigaciones en relación al índice de educación. Se describen las bases teóricas de la investigación, segmentadas en diseño del videojuego, los beneficios latentes de los videojuegos en temas de la educación del idioma y su influencia en general en otras ramas aparte de la educación y de su importancia actual.

El tercer capítulo desarrollará la metodología de la investigación, Aquí se especifica el modo del diseño del videojuego. Para ello, se hará énfasis en las técnicas necesarias para desarrollo del videojuego, considerando temas como la interacción entre el juego y el usuario, además de temas de diseño más enfocada en la retención de atención del usuario.

En el cuarto capítulo se hace el análisis de los resultados obtenidos por la realización del proyecto en relación con las hipótesis planteadas.

Finalmente, en el último capítulo, se discuten los resultados del proyecto, se resaltan las cualidades del videojuego en general y de cómo estos ayudan a enfrentar los problemas actuales de la educación actual.

Capítulo I: Planteamiento del problema:

El presente capitulo describe la problemática que existe actualmente en el ámbito educativo, así como su relación con la aplicación de un mejor diseño en relación a la programación de juegos serios en relación al tema de educación del vocabulario del idioma inglés. Se hace relación a este problema para así enfrentar el gran problema de educación actual en el Perú, el cual fue posicionado en último lugar en relación al desempeño del alumnado en temas académicos como matemáticas, lectura y ciencias, entre otras enseñanzas (Andean Air Mail & Peruvian Times, 2013).

1.1 Descripción de la Realidad Problemática

Según el estudio realizado por el British Council of Educational Intelligence, aunque el Perú ha logrado una educación primaria casi universal y ha expandido la educación secundaria sin aumentar significativamente su presupuesto, el cual se ha logrado al expandir el sector de educación privada. Sin embargo, la mayoría de comunidades indígenas multiétnicas tienden a tener menor acceso a este tipo de educación, resultando en un desequilibrio educativo y socio-económico. La mayoría de los estudiantes peruanos estudian inglés en la escuela secundaria (57%), mientras el 46% aprendió durante sus estudios universitarios y el 41% asistió a escuelas privadas de idiomas. En relación a la motivación por el aprendizaje, un 44% aprendió el idioma porque era obligatorio en la escuela secundaria mientras que el 40% aprendió porque era necesario para la educación universitaria. Además de todo ello, el aprendizaje del idioma inglés es considerado como un commodite, ya que requiere alta inversión de costo y tiempo. (British Council of Educational Intelligence, 2015)

La política educativa ha evolucionado y variado constantemente por un contexto de inestabilidad política. Estos y diversos otros factores, los cuales tienen que ver con temas de falta de presupuesto, falta de docencia de credibilidad, han creado una calidad educativa que resulta ser preocupante para el Perú, el cual obtuvo el puesto 65 de 144 países para la educación primaria en el Informe de Competitividad Global 2014, con resultados deficientes para matemáticas, ciencias, educación superior y capacitación, y el último lugar en el ranking PISA 2012 de la OCDE (Organization for Economic Co-operation and Development), que compara el desempeño de los estudiantes de secundaria en

matemáticas, lectura y ciencias, en donde los estudiantes provenientes de entornos vulnerables tienen las puntuaciones más bajas, resaltando el desnivel previamente explicado (Andean Air Mail & Peruvian Times, 2013).

Al mismo ritmo del desarrollo de nuevas tecnologías de tipo computacional y avances en la medicina, se ha explorado también nuevas maneras mundiales de enseñanza de lenguaje. Richards y Rodgers afirman que existen métodos de aprendizaje de lenguaje, como el CLT (Communicate Language Teaching) o enseñanza por medio de comunicación, el cual es la manera más común de enseñanza en la actualidad, con la ventaja del mejor conocimiento de entonaciones y fluencia de habla; CBI (Content-Based Instruction) o instrucción basada en contenido, consistiendo del uso de herramientas, sean virtuales o físicas, para aprender el lenguaje objetivo, la cual es muy utilizada en Estados Unidos para preparar a sus estudiantes a entrar a educación de nivel primaria y secundaria. Finalmente, CLIL (Content and Language Integrated Learning) o aprendizaje integrado de contenido y lenguaje, el cual es similar al método CBI, involucra una mezcla entre el contenido y el lenguaje para fomentar la educación, el cual es popular en partes de Europa para estudiantes de temprana edad. (Richards, Jack . C., & Rodgers, Theodore. S., 2014).

Sin embargo, incluso con todas los avances y métodos de enseñanza en general, no resulta efectiva la enseñanza si los alumnos en cuestión no muestran interés en el tema o simplemente no están conformes con las maneras actuales de enseñanza. Profesores y docentes de universidades, como respuesta a este problema, se han dirigido a la industria de los videojuegos como una manera de solucionar el problema de la atención de sus alumnos.

Los videojuegos son parte de la vida de los niños y adolescentes (además de personas de oficina, en el caso de juegos casuales), con un 97% jugando al menos un juego por hora en los Estados Unidos. (Granic, Lobel y Engels, 2014). En la actualidad, en los Estados Unidos, 91% de los niños en un rango de edad entre 2 y 17 años juegan videojuegos y un estudio nacional de los adolescentes determinó que un 99% de varones y 94% de mujeres juegan estos juegos. En comparación, aquí en Perú, existen 7.2 millones de personas que se dedican a este hobby de un total de 31.38 millones de personas. Eso quiere decir que el 23% de la población juega videojuegos. Además, afirman que, al contrario de creencias populares de que los videojuegos son intelectualmente bajos en contenido y usado como sedante del cerebro, se ha identificado que estos juegos

promueven no solo mejor atención, sino una gama de habilidades cognitivas (procesamiento visual, habilidades relacionados a resolución de problemas fuera o dentro de un contexto, etc.). (Granic, Lobel y Engels, p.68, 2014).

Una infinidad de reseñas existen en relación a los efectos y éxitos de aprendizaje asociados con el uso de juegos como herramientas educativas (O'Neil, Wainess, & Baker, 2005), y un meta análisis concluyo que los juegos pueden hacer importantes avances en la reforma educacional para tratar con los retos educacionales del siguiente siglo. (Vogel, 2006).

Cualquier tipo de niño o adolescente ha tenido por lo menos contacto con un videojuego, el cual posee no solo cualidades de entretenimiento, sino también para la educación. Instituciones como el SGI (Serious Game Institute) en el Reino Unido, promueven la educación por medio del uso de juegos serios para educar a sus alumnos en temas como ingeniería, física y otros rubros de nivel universitario, con un éxito favorable.

Para mejorar dicha situación referenciada, se planea crear un juego educativo con el propósito de mejor la enseñanza del idioma inglés en el tema del vocabulario, que logre captar la atención del usuario para que pueda aprender de manera lúdica de una manera más efectiva del resto de juegos educativos de la actualidad que están en circulación en otros países con E.E.U.U. El rubro de enseñanza por medio de los videojuegos en muy joven en el Perú, y se espera estar en la vanguardia de este nuevo impulso en relación a los videojuegos.

1.2 Formulación del problema:

1.2.1. Problema general:

¿Es el aprendizaje mediante videojuegos efectivo para la enseñanza del vocabulario en inglés para niños de un rango de edad entre 5 y 11 años?

1.2.2 Problemas específicos:

¿Qué elementos se tienen que tomar a consideración en el diseño del juego para que sea educativo?

¿Qué elementos educativos se quieren plasmar en el juego?

¿Qué tanto porcentaje de personas han mejorado su vocabulario de inglés después de probar el prototipo?

1.3 Objetivos de la investigación:

1.3.1 Objetivo general:

Definir los factores que justifican la efectividad de la enseñanza del vocabulario en inglés para niños de 5 a 11 por medio de un videojuego.

1.3.2 Objetivos específicos:

Determinar los elementos que se tienen que considerar para el desarrollo del juego en relación al objetivo educativo.

Delimitar los elementos y/o temas a agregar en el juego para lograr el objetivo educativo.

Precisar el porcentaje de personas que han logrado mejorar su vocabulario de inglés al jugar nuestro juego.

1.4. Justificación:

1.4.1 Ámbito teórico:

En el ámbito teórico, la exploración de los videojuegos como herramientas de aprendizaje ha tenido antecedentes exitosos en diversos países del primer mundo. Yolageldili, G. y Arikan, A. (2011) exploraron la efectividad del uso de los videojuegos para enseñar a alumnos a aprender gramática. Los alumnos muestran ansiedad al ser criticados o castigados por sus profesores cuando cometen un error. La ventaja de los juegos en este ámbito proviene de que reducen ansiedad e incrementan sentimientos positivos y de autoconfianza, facilitando la aceptación de nuevos conocimientos. Profesores en la actualidad ven a los videojuegos como herramientas indispensables y cruciales para la enseñanza del idioma anglosajón en las escuelas primarias ya que proveen a los profesores con beneficios educacionales que no eran posibles con otras herramientas, como son la captación de atención del alumno.

Para ello, con el videojuego se busca la mejora de la educación de los alumnos peruanos del idioma inglés por medio de la interacción del alumno con el videojuego, el cual presentará objetivos para que el alumno logre aprender nuevas terminologías de vocabulario y escritura mientras este, a la vez, se entretenga con el contenido presentado, maximizando la calidad de aprendizaje.

Se demuestra que: profesores y alumnos ven a los videojuegos como herramientas efectivas de enseñanza de vocabulario; la mayoría de los alumnos prefieren juegos en línea como una ayuda educacional en comparación a lecciones de aprendizajes tradicionales; y, para incrementar el interés de los alumnos y garantizar la efectividad del aprendizaje, se requieren de juegos más motivadores que brinden a los alumnos un sentimiento de logro y de auto mejora. Cabe destacar que los juegos cada vez se vuelven más sofisticados con el paso del tiempo, y esta expectativa aumenta el estándar de calidad que deben presentar los juegos para que sean aceptables por el público en general. Finalmente, los videojuegos están muy relacionados con otro beneficio cognitivo: incremento en la creatividad cognitiva. Nuevas evidencias demuestran que, jugar cualquier juego, sea o no violento, incrementa las capacidades creativas de los niños, lo cual conlleva a una asociación directa

con la resolución de problemas complejos o que requieran un diferente tipo de mentalidad. (Yip, F.W.M., & Kwan, A.C.M.(2006)).

1.4.2. Ámbito práctico:

Como aporte practico, el proyecto de videojuego brindara una mejor manera de educación del inglés por medio de la interacción entre la interfaz del juego y el alumno, tomando en consideración las ventajas que posee el juego en relación a los textos didácticos, como lo son la captación de atención del alumno, la facilidad de distribución de los juegos y del sentimiento de logro al aprender algo el cual motiva al alumno a desempeñarse mejor.

En el estudio realizado por Hamari, J., Shernoff, D.J., Rowe, E., Coller, B., Asbell-Clarke, J. y Edwards, T. (2016), la atención concentrada es una ocurrencia simultánea de concentración elevada, interés y entretenimiento, encapsulando el tema de la experiencia educativa. Estudios anteriores de Hsu, C.-Y., Tsai, C.-C. y Wang, H.-Y (2012) y Akkerman, S., Admiraal, W., & Huizenga, J. (2009) han demostrado una positiva asociación entre entretenimiento y aprendizaje, y además de que el entretenimiento y atención en el juego puede resultar en concentración para mayor ratio de notas y aprendizaje. (Tüzün, H., Yilmaz-Soylu, M., Karakus, T., Inal, Y., & Kizilkaya, G. 2009).

El campo de utilización de los videojuegos como herramienta de educación es una tendencia novedosa aquí en el Perú. Por lo tanto, se afirma que la creación y aplicación del videojuego planeado en cuestión solucionaría varios problemas en relación del nivel disparejo de educación lingüística. Por un lado, los videojuegos pueden ser diseñados con un solo tema a enseñar en mente, lo cual facilita en el desarrollo del mismo. Además, con el avance actual de la tecnología, se es posible transmitir el videojuego por medio de Internet a las zonas rurales, sin la necesidad de colocar presupuesto en el transporte de los mismos, una ventaja clara en comparación con el transporte de libros acerca del tema. En sí, se espera mejorar la calidad educativa, removiendo su característica de commodite imposible de acceder para las familias con pocos recursos en sí. Al estimarse que las escuelas actuales poseen por lo menos un salón de cómputo, hacer llegar el videojuego educativo a las poblaciones más necesitadas no resultara ser ningún problema.

1.4.3. Ámbito metodológico:

En el ámbito metodológico, se espera, por medio de la elaboración del videojuego para la enseñanza del idioma inglés en base del contexto peruano, mejorar el nivel actual de educación de los alumnos de zonas rurales de provincia, en los cuales el nivel educativo es menor del estándar actual.

Además, se espera que, con la demostración de la aplicación del videojuego como herramienta de educación aquí en el Perú, generar una mejora en las políticas generales en relación a los métodos de educación actual para el beneficio de varios jóvenes en relación a su aprendizaje en el futuro cercano. El proceso de la creación del videojuego en sí involucrara la aplicación de métodos de informática computacional, con el fin de dar un contenido sustancial al videojuego como herramienta completa para la educación peruana.

1.5 Delimitación del estudio:

Como se ha detallado antes, se pretende realizar una evaluación del prototipo del videojuego con propósitos educativos dentro de áreas de Lima Metropolitana con nivel económico de tipo C a menor, con el propósito de analizar los resultados del videojuego en sí en relación a la población demográfica.

El desarrollo post-análisis consistiría en analizar el *feedback* en relación a la aceptación general del videojuego. Asimismo, se analizará detalles como atención captada de los participantes y cantidad del material aprendido en relación a las sesiones tomadas en cuenta. Finalmente, la distribución en sí del videojuego será de medio digital (o virtual), tomando en cuenta de las facilidades naturales que poseen los videojuegos en términos de distribución.

Si las pruebas en dicho ambiente resultan ser positivas, se presentara la idea de establecer otras ideas de videojuego, abarcando otros temas educativos como ciencias, matemáticas, física, con el propósito general de mejorar el nivel general de educación y además de balancear la gran diferencia educativa entre Lima y las provincias, para así evitar conseguir un ratio de educación bajo en comparación del resto de América Latina, cambiando la posición actual del Perú de manera positiva en no solo el ámbito educativo, sino a un nivel social-competitivo.

Capitulo II: Marco Teórico.

2.1. Marco de antecedentes del estudio:

Los juegos constituyen un dominio de estudio que atrae el interés de científicos de varias disciplinas. Esto es particularmente cierto desde la perspectiva de la inteligencia computacional. Para ello, en orden de examinar la creciente importancia en esa área en el territorio de los juegos, se ha realizado un análisis de la colaboración científica de investigadores trabajando en el tema de inteligencia computacional en videojuegos (CIG). El estudio de la red de CIG se ha realizado por medio de una perspectiva temporal a nivel macroscópicos, mesoscopicos y microscópicos, estudio la estructura a larga escala, las mecánicas de crecimientos y los patrones de colaboración entre otros factores. En general, la inteligencia computacional en videojuegos exhibe similitudes con otras redes de colaboraciones. La evolución macroscópica y mesoscopica de la red indica que el campo es altamente activo y vibrante, pero aún se encuentra en sus fases iniciales. (R.Lara-Cabrera & Cotta, 2014)

Según lo analizado por los investigadores en relación a la inteligencia computacional de los videojuegos, se ha descubierto que el campo es de alto interés, atrayendo la atención de varios autores e investigadores. Aunque el campo investigativo esta aun en fases tempranas de desarrollo, muestra un robusto crecimiento para varias investigaciones relacionadas al tema.

En relación a la información de la investigación, se observa que varias tendencias están presentes. La Figura 1 muestra un pequeño fragmento de la red total y cómo ha evolucionado a través del tiempo. La curvatura muestra el incremento positivo del número de autores desde los 2000 hasta el 2010, donde el número se estabiliza. Esto indica que el tema de inteligencia computacional en los juegos es un activo y vibrante campo que cada vez atrae la atención a nuevos investigadores al campo y generando nuevos reportes e investigaciones.

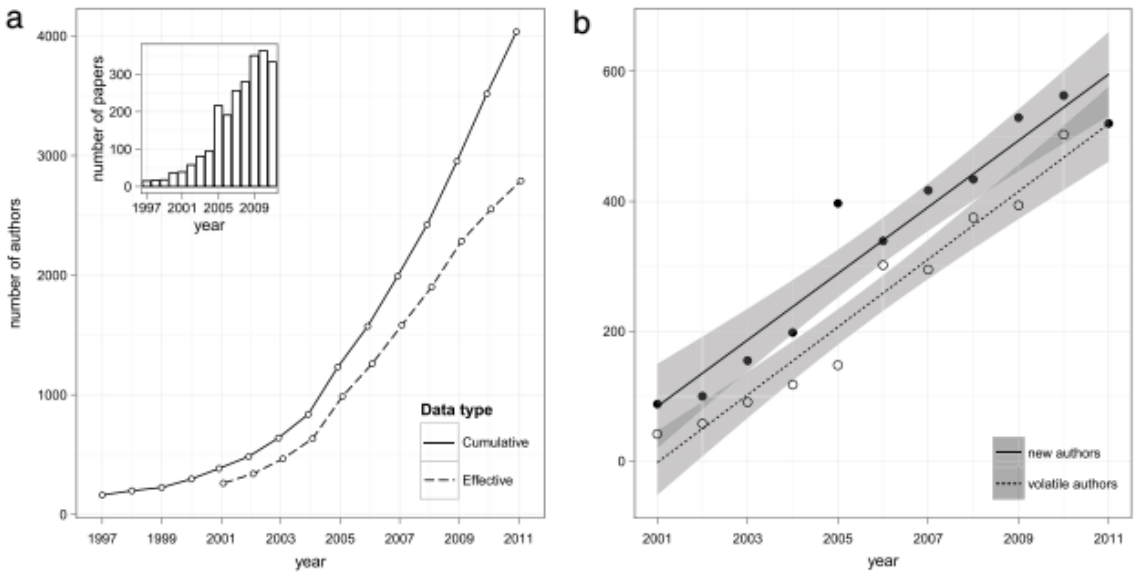


Figura 1: (a) Evolución del total de autores que publican reportes de temas de videojuegos al año. (b) Evolución temporal del número de nuevos autores y autores volátiles (información anual). Las líneas rectas son los ajustes lineales (Δx = 51.15, R2 = 0.92 para nuevos autores y Δx = 52.16, R2 = 0.95 para autores volátiles.)

Fuente : R.Lara-Cabrera, Cotta (2014) *An analysis of the structure and evolution of the scientific collaboration network of computer intelligence in games*, 395, 523–536

La industria de los videojuegos es una fuerza poderosa que alienta la evolución de la tecnología de computadoras. Debido que las capacidades de las computadoras personales, hardware periférico y consolas de videojuegos, igual ha aumentado la demanda por la calidad de información acerca de los algoritmos, herramientas y descripciones necesarias para tomar ventaja de esta nueva tecnología. Para satisfacer dicha demanda y establecer un nuevo nivel de profesionalismo al desarrollo de videojuegos, el libro en sí hace referencias a la creación de la serie de libros de Morgan Kaufmann acerca de la interactividad de tecnología 3D. Estos libros están escritos por desarrolladores liderando la industria e investigadores académicos, cubriendo el tema de la aplicación de 3D. La serie referencia a soluciones y principios de ingeniería de software sólidos y prácticos. (Millington, I., 2006).

Según Yoke S., Wong y Hayati, M (2014), los juegos de computadora han sido adoptados como parte de las herramientas de aprendizaje y enseñanza en el ambiente. Aunque aún perduran las creencias del negativo impacto de los mismos debido a que su

uso puede causar adicción en los estudiantes, algunos estudios muestran que, usando el mismo método tradicional de educación incluyendo a los juegos de computadoras como herramientas de enseñanza y aprendizaje, los estudiantes pueden aprender habilidades y conocimientos de manera más eficiente. Por ello, los juegos de computadora poseen un gran potencial para asistir a profesores y motivar a los estudiantes en una manera nueva y retadora. El propósito general del estudio realizado fue de identificar y evaluar como los juegos de computadoras puede ser aplicado como herramientas de aprendizaje y enseñanza para enseñar programación orientada a objetos en una institución de alta enseñanza. En actualidad, los juegos de computadoras han sido aplicados en el proceso de aprendizaje en el campo educativo debido al incremento del uso de las computadoras en el mercado.

Anteriormente, la percepción de los juegos de computadoras sostenía que estos estaban más dirigidos para niños, pero la gente se dio cuenta que esta percepción era errada; no solo los niños requieren entretenimiento, los adultos también. Adultos pueden aliviar sus tensiones y estrés por medio de la inmersión al mundo digital de los videojuegos. Aparte de ello, por medio del uso de las computadoras como herramienta de aprendizaje, las capacidades de aprendizaje y entendimiento del estudiante se verán incrementados. Esto se debe a que los juegos de computadora educativos proveen al estudiante un contexto de interacción. Estudiantes pueden interactuar con el juego y recibir feedback inmediato acerca de su desempeño. Por ello, este estudio busca llenar el vacío en investigación acerca del uso de los juegos de computadora como una herramienta de enseñanza orientada para programación orientada a objetos en institución de alto aprendizaje en Malasia.

En sí, el estudio provee una oportunidad para los estudiantes para participar en el proceso de aprendizaje por medio de la interacción con las funciones de los juegos de computadora. Esto se debe a que las funciones y/o mecánicas de un juego de computadora son elementos importantes al diseñar un juego de computadora, además de ser fáciles de recordar por los jugadores. Elementos como el Game Flow (Figura 2) y la interacción entre elementos de los mismos juegos muestran una manera de educar a estudiantes por medio de asociación de los mismos elementos a situaciones o elementos de otros temas o contextos.

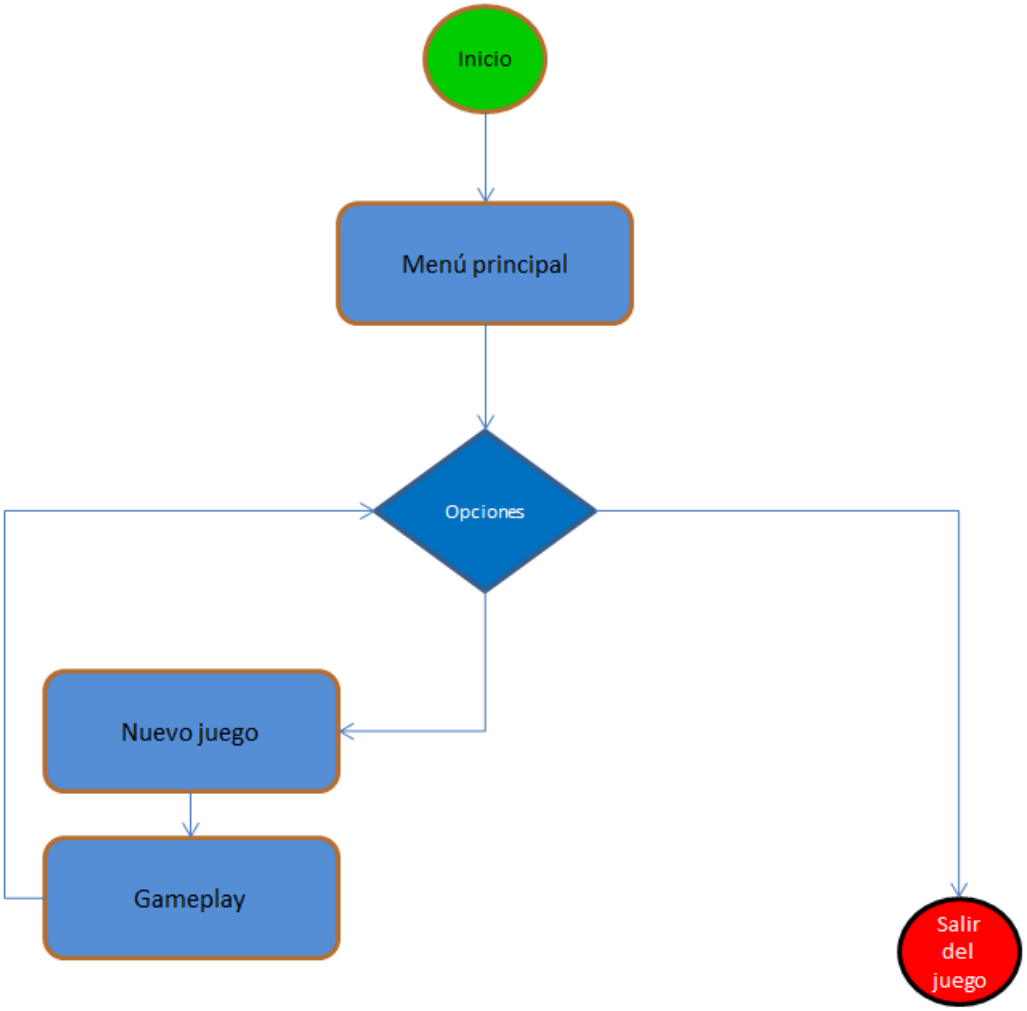


Figura 2: Game Flow (elaboración propia)

Según el estudio presentado por Ashrafa, H., Ghanei, F. y Salamie, M (2014), se reporta la utilidad de los juegos en línea para el aprendizaje EFL (English as a Foreign Language) para estudiantes iraníes. Se asignó a 24 alumnos de desempeño bajo-intermedio de manera aleatoria a un grupo experimental y otro de control. El grupo experimental aprendió vocabulario por medio de un juego de computadora en línea por 15 semanas. Para medir los resultados, se realizó pruebas escritas basadas en vocabulario tanto antes como después de la prueba a ambos grupos. Como resultado, se comprobó que el grupo que jugo el juego para aprender vocabulario supero en desempeño en una gran magnitud al otro grupo. Por ello, se determinó que el uso de videojuegos en línea es más efectivo en la enseñanza del vocabulario inglés.

Según lo demostrado, aquí se presenta un buen ejemplo de la aplicación de educación en los videojuegos para impulsar el aprendizaje de los alumnos por medio de los juegos, dependiendo del contexto al que se aplican.

Los videojuegos han estado alrededor por casi 50 años. Recientes descubrimientos científicos muestran que los contenidos de los juegos con contenidos pro-sociales tienen gran potencial para incrementar la calidad de vida de los niños y adolescentes, pero exposición ante juegos anti-sociales o violentos tiene un gran porcentaje de generar resultados negativos, cuyo riesgo aumenta mientras más exposición exista hacia estos juegos. (Anderson, A. A., Warburton, W. A., 2012).

En el estudio realizado por Salceanu, C. (2014), se busca investigar la posición de los padres en relación a la influencia de los juegos de computadora en el desarrollo de los jóvenes tomando en cuenta los siguientes aspectos: el tiempo consumido al jugar, tipos de juegos favoritos, maneras de supervisión por parte de los padres y beneficios y desventajas de los videojuegos (desde el punto de vista de los padres). Los resultados muestran:

- 30.47% de los niños pueden acceder a la computadora cada vez que quieran.
- La computadora es más usada para juegos (36.28%)
- 42.87% de los padres supervisan a sus hijos solo cuando tienen tiempo libre.
- 50% de los padres permiten a sus hijos consumir de 1 a 2 horas en la computadora cada día mientras que un 28.54% permite de 3 a 4 horas.

Las más notables ventajas de los juegos de computadora, identificadas por los padres, son el desarrollo del pensamiento (9.60%), capacidad de observación (8.27%) y creatividad (8.01%). Las más notables desventajas han sido la falta de movimiento físico (13.37%), desordenes de vista (13.15%) y agitación (8.58%). Sin embargo, los padres reconocen el potencial de los juegos y sus notables efectos sobre los niños, por ello un control de la cantidad y contenido de los mismos es necesario.

Ferguson, C.J. y Olson, C. (2013) muestran que “la gran mayoría” de investigación psicología en relación al “gaming” han sido enfocados en sus aspectos negativos: el potencial daño en relación al incremento en agresión, adicción y depresión. Los autores opinan que para entender el impacto de los videojuegos en adolescentes y niños se es necesario tener una perspectiva más balanceada. Considerando los beneficios potenciales son importantes en parte, debido a la naturaleza de estos juegos cambió de manera

dramática la última década, volviéndose más complejos, diversos, realistas y sociales en naturaleza.

Green, C. S. y Bavelier, D (2012) muestran que jugar juegos promueve una gran variedad de habilidades cognitivas, como, por ejemplo: mejor y más rápida atención en relación a problemas de asignación, mayor capacidad de procesamiento visual y aumento en habilidades de comprensión mental. Por ejemplo, según lo que los autores plantean, un juego de acción se distingue de otros géneros por la velocidad de sus juegos, debido a que tienen objetos transigentes que aparecen dentro y fuera del rango de visión del jugador de manera acelerada, lo cual fuerza a los jugadores a constantemente realizar predicciones en relación a eventos futuros dentro del contexto del juego en un tiempo de atención muy corta. La habilidad de mantener atención en un objetivo e ignorar información distractora es la esencia de atención selectiva. Correspondientemente, los resultados de gameplay en juegos de acción resultan en una mejor localización de objetivos, como demostrado en el perímetro de Goldman, o entre información distractora o irrelevante, como se ve en un campo de visión útil o en una búsqueda visual estándar. (Figura 3).

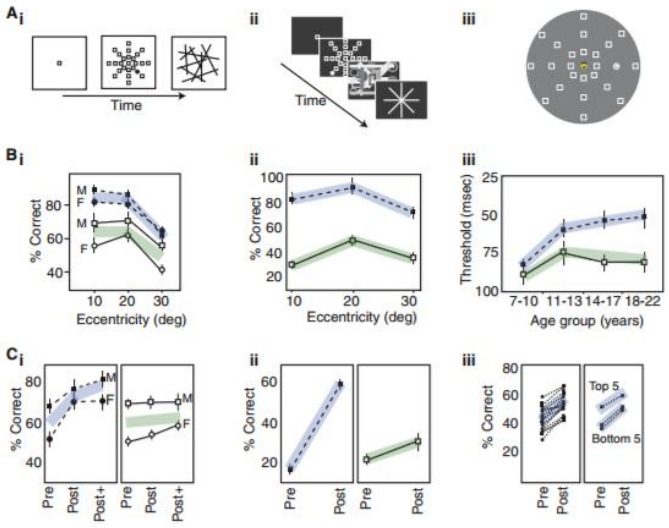


Figura 3: Mejora de la atención espacial selectiva después de gameplay de juegos de acción, sea por mejora de visión(A), resolución de objetivos (B) y desempeño en el entrenamiento (C).

Fuente: Green y Bavelier (2012) *Learning, Attentional Control, and Action Video Games*

2.2. Bases Teóricas:

2.2.1. Aprendizaje basado en videojuegos:

2.2.1.1 Descripción:

En la actualidad, el rubro de aprendizaje por medio de los videojuegos ha empezado a tomar campo en estos últimos años. Hamari, J., Shernoff, D.J., Rowe, E., Coller, B., Asbell-Clarke, J. y Edwards, T. (2016), en un contexto de juego ideal para la educación, los estudiantes aprenden a cómo resolver problemas complejos, debido a que los problemas a resolver en un juego típicamente empiezan con un nivel de dificultad bajo y progresivamente incrementan en dificultad. Los videojuegos brindan conocimientos y experiencias relacionadas al rol o meta planteada por el mismo, volviendo su enseñanza tanto explicita como implícita en ciertos casos.

2.2.1.2. Ventajas generales a la enseñanza con videojuegos:

Griffths, M. (2002) afirma acerca de algunas ventajas claves del aprendizaje basado por videojuegos consiste en:

- Los videojuegos pueden ser usados como herramientas de investigación y/o desarrollo. Además, como herramientas de investigación, estas poseen una gran diversidad.
- Los videojuegos atraen la participación de varios individuos de diferentes dimensiones demográficas (adultos, ancianos, etc).
- Los videojuegos pueden ayudar a los niños a definir metas, proveer feedback inmediato, refuerzo y a mantener archivos acerca de sus cambios emocionales.
- Los videojuegos permiten al investigador medir desempeño por medio de una serie de tareas, y puede ser libremente cambiada, estandarizada y entendida.
- Los videojuegos pueden ser usados cuando se examinan características individuales como autoestima, autoconcepto, definición de metas y diferencias individuales.

- Videojuegos son divertidos y estimulantes para los participantes. Consecuentemente, es más fácil lograr y mantener la atención de una persona por largos periodos de time.
- Los videojuegos pueden proveer elementos de interactividad que puede estimular el aprendizaje.

2.2.1.3. Beneficios educativos en relación a los videojuegos:

2.2.1.1.1. Beneficios cognitivos:

Según Granic, I., Lobel, A., & Engels, R.,C.,M.,E. (2014), contrario a las creencias en relación a jugar videojuegos que vuelven a sus usuarios intelectualmente flojos, se ha descubierto que jugar estos juegos promueve una gran variedad de habilidades. Tomando por ejemplo a jugadores de juegos de genero *shooter*, como *Half-Life,* se ha descubierto que, a comparación a otro grupo de personas que nunca han jugado un juego *shooter*, los jugadores veteranos de este género muestran una atención espacial más ágil y enfocada, mejor resolución espacial en relación al proceso visual y una mejora a las respuestas de situaciones fugaces.

Se ha investigado que estas habilidades espaciales pueden ser entrenadas en cualquier persona en un breve periodo de tiempo y que estos beneficios duran por un extendido periodo de tiempo, y, crucialmente, estas habilidades se pueden también en un contexto real fuera del mismo juego. Estas ventajas se manifiestan de manera medible en el proceso y eficiencia neural; por ejemplo, se determinó que, por medio de un escaneo de resonancia magnética, zonas del cerebro dedicadas a la atención localizada eran menos activas en *gamers* en comparación a personas que no juegan juegos regularmente en actividades de detección de patrones, lo cual sugiere que los *gamers* utilizan mejor sus recursos de atención y filtran mejor la información presentada en relación a la resolución de retos de alta complicación

2.2.1.1.2 Beneficios motivacionales:

Granic, I., Lobel, A., & Engels, R.,C.,M.,E. (2014) afirman que para que los beneficios generales de los videojuegos se apliquen en los usuarios de los mismos, se requiere que los juegos capten la atención del usuario. Existen características clave de como los videojuegos promueven una motivación en general dentro y fuera del contexto de los juegos; por ejemplo, los videojuegos demuestran *feedback* inmediato de los esfuerzos realizados por los usuarios, desarrollando en los usuarios un sentimiento de satisfacción y automejora para no repetir los mismos errores cometidos durante su sesión de juego.

Feedback concreto e inmediato sirvió como una herramienta para premiar los esfuerzos continuos de los jugadores y así mantenerlos en una zona de confort que balancea óptimos niveles de reto y frustración con suficientes experiencias de éxito y desarrollo. El sentido de fracaso, en este entorno, en vez de ser un aspecto negativo que limita al jugador, como se ha determinado que la inteligencia o habilidad es una marca de esfuerzo propio, las señales de falla denotan la necesidad del jugador de autosuperarse. En consecuencia, esta mentalidad positiva conlleva a una mejora en el desempeño académico.

Resumiendo, si es que jugar videojuegos puede ser considerado un mal pasatiempo, con resultados distractivos, este mismo también es un buen ambiente para cultivar un estilo motivacional de auto mejora que puede ser aplicado en contextos de trabajos y/o actividades escolares.

2.2.1.1.3. Beneficios emocionales:

Granic, I., Lobel, A., & Engels, R.,C.,M.,E. (2014) afirman que en la actualidad, nosotros como seres humanos utilizamos varios medios como cine, arte entre otros para distraernos y mejorar nuestro esto emocional, de los cuales los videojuegos son el medio más eficiente y efectivo de generar reacciones positivas, tanto en niños, como en jóvenes adultos. Sentimientos como orgullo o logro en relación al superar retos difíciles son una de los sentimientos que muchos de los *gamers* activamente buscan para su bienestar emocional. Sin embargo, los videojuegos no contienen en su totalidad sentimientos positivos. Durante una sesión de juego, el jugador puede sentir sentimientos negativos como frustración, ira, ansiedad e incluso tristeza. Sin embargo, como sostiene Gottman, J. M. (1986), el contexto simulado de los videojuegos puede ser lo suficientemente real para volver los logros realizados en los mismos que tengan un mayor significado de lo normal pero también lo suficientemente seguro para controlar o modular dichas emociones negativas para que sirvan más para la resolución de las metas establecidas por el videojuego.

2.2.2 Aprendizaje de vocabulario inglés:

Actualmente, la enseñanza del vocabulario en inglés es un aspecto básico para los niños. Se estima que la edad óptima para aprender inglés con facilidad como primera lengua es después de los 3 años. Pasado este rango, los niños aprenden inglés como una segunda lengua. Mientras mayor sea la edad del aprendiz, mayor será la dificultad del aprendizaje del idioma.

Contemporáneamente, se han realizado varios estudios sobre el aprendizaje de inglés por edades. A la edad de 3 años el niño va incluyendo inflexiones morfológicas en su diálogo y, conforme avanza en su aprendizaje, va aprendiendo y aplicando más sintaxis del idioma.

2.2.2.1 Obstáculos en el aprendizaje del idioma inglés en el Perú:

Según un estudio realizado por el British Council of Educational Intelligence, existen ciertos obstáculos en relación al aprendizaje del idioma inglés, el cual se ha vuelto un idioma necesario en temas de negocio:

- Para la mayoría de la población, el aprender inglés es un lujo que solo puede ser enseñado en la mayoría de los casos en institutos.
- En general, la calidad educativa del Perú ocupa uno de los rangos más bajas, según el ranking PISA del 2014.
- Hay una escasez de maestros correctamente capacitados para dictar clases de inglés en general.

En el mismo estudio (Figura 4), se había realizado una encuesta hecha a 501 participantes que no habían aprendido inglés debido a sus experiencias con el lenguaje, y se revelo la siguiente información, brindada en la siguiente tabla:

Figura 4: Razones por las cuales usuarios no quieren aprender inglés

Fuente: British Council Educational Intelligence, Base de Datos para America Latina, 2015

Las razones más importantes para no aprender inglés se relacionaban estrechamente con el costo y el acceso: casi la mitad de los encuestados (47%) informaron de que aprender inglés era demasiado caro, mientras que casi un tercio (32%) dijo que no tenía tiempo y más de una cuarta parte (26%) mencionó la falta de acceso a los programas financiados por

el gobierno. Además, uno de cada cinco encuestados no había aprendido porque sentían que no eran buenos para aprender idiomas, mientras que otros porcentajes indicaron que no estudiaron inglés en la primaria (17%) o (16%) en la secundaria. Pequeños pero significativos porcentajes no habían aprendido porque no querían (7%) o porque el inglés no era necesario para su trabajo (3%).

2.2.2.2 Morfología:

La adquisición de la morfología del idioma inglés varía a través de las edades de los alumnos. A medida que su edad avanza, la sintaxis del niño va cambiando y haciéndose más compleja. En la Tabla 1 se presenta la evolución por edades:

Aspectos Morfológicos	Edad Media de Obtener/Aprendizaje
Ing (playing)	3 / 5 años
Plural regular	3 / 5 años
Plurales irregulares	4 / 6 años
Pasado Verbos Irregulares	5 / 7 años
Pasado Verbos Regulares	6 / 8 años
S. Genitivo	7 / 8 y etapas posteriores
3ra. Persona, presente singular	8 años y etapas posteriores

Tabla 1: Tabla de tiempo de aprendizaje por morfología

Fuente: Moya, A. J. & Jimenez, M.J. *El proceso de interlengua en el aprendizaje del inglés como lengua extranjera en edades tempranas* (2004)

2.2.2.3 Metodología de enseñanza de lenguas extranjeras:

Según Reynoso, H. (2000), en la enseñanza de lenguas se distinguen entre métodos generales y específicos, tradicionales y contemporáneos; más sin hacer válidas estas distinciones -entre otras razones por vagas e imprecisas-, la literatura ha documentado un sinnúmero de métodos que ha clasificado de acuerdo a: categorías lógicas (síntesis, análisis, inducción, deducción); el aspecto de la lengua en el que centra su atención (gramatical-léxico, fonético, etc.); las habilidades que se entrenan (traducción, oral, escrito, de lectura); la teoría base lingüística o psicológica del aprendizaje en la que se

apoya (consciente, sugestopédico, estructural, etc.) y también se les llama de acuerdo a su inventor o figura más prominente (el método de Comenius, Gouin, Berlitz, Palmer, Lozano, Jorrín, etc.).

Existen actualmente diferentes métodos para la enseñanza de un idioma extranjero:

- Metodología natural: El proceso de aprendizaje de una segunda lengua debe realizarse de la misma forma que en el caso de la lengua materna.
- Traducción gramatical: Se presta atención a la asimilación de las reglas gramaticales a través de la presentación de una regla, estudio de una lista de vocabulario y ejecución de una lista de traducción.
- Respuesta física total: Combina el habla con la acción y proponer enseñar la lengua a través de la actividad física.
- Silencioso: Reducción drástica del habla del profesor y un gran énfasis en las producciones del alumno.
- Audiolingual: Práctica sistemática de la lengua oral donde la repetición oral o a través de ejercicios era la forma de establecer hábitos lingüísticos en la nueva lengua.
- Sugestiopedia: Desarrolla una serie de técnicas para conseguir la motivación de los alumnos.

La básica premisa de Berlitz (el método directo) consiste en que el aprendizaje de segunda lengua debe ser más como el aprendizaje de la lengua primaria: mayor interacción de actividad oral, uso espontaneo del lenguaje, sin translación entre los dos lenguajes y poco o ningún análisis del vocabulario o reglas gramaticales. (D.H., 2007) Los principios del método directo están resumidos en:

- Instrucción dentro del aula era conducida exclusivamente en el lenguaje que se deseaba aprender.
- Solo vocabulario y oraciones del cada día era enseñadas.
- Comunicación oral era cementada mediante una progresión organizada basada en conversaciones pregunta-respuesta entre profesores y alumnos.
- Gramática era enseñada inductivamente.
- Nuevos términos de enseñanza eran introducidos oralmente.

- Vocabulario concreto era enseñado mediante demostración, objetos y figuras; vocabulario abstracto, por medio de interrelación de ideas.
- Compresión auditiva y oral también eran enseñadas.
- Correcta pronunciación, enunciación y gramática era enfatizada.

El método directo de enseñanza disfruta popularidad actual, siendo mayormente aceptada en escuelas privadas de lenguaje, donde los estudiantes se encuentran mejor motivados en el aprendizaje si el profesor es de nacionalidad del idioma objetivo.

2.2.2.4 Metodología de aprendizaje de vocabulario inglés:

Según Blázquez, A. (2010), se señala seis métodos distintos que han sido utilizados hasta el momento. El primero de ellos es el método clásico o gramatical en el que se da más importancia a la lengua escrita que a la lengua hablada y se basa en el aprendizaje de reglas gramaticales y listas de palabras. La traducción directa e inversa es otra de las características de este método.

El método analítico o interlineal. Este método se basaba en el uso de un texto literario escrito en la lengua que se iba a enseñar y a partir de él se hacían traducciones y se aprendían palabras o estructuras gramaticales utilizadas en el mismo.

El método directo con el que se introduce una nueva orientación a la enseñanza de los idiomas. Si en los dos anteriores el aprendizaje del vocabulario estaba basado en la memorización de listas de palabras, ahora este vocabulario se aprenderá mostrando objetos y situaciones en los que sea utilizado. Por otra parte, la gramática no se aprenderá como reglas aisladas sino que su aprendizaje buscará facilitar una mejor comunicación. Otro de los cambios importantes es que en la clase no se utiliza la lengua materna. De esta nueva forma de entender la enseñanza de una lengua extranjera, que surge a finales del siglo XIX, derivarán otras nuevas en el siglo XX como son:

- *Método audio-oral.* En el que la expresión oral será la base del aprendizaje que se basará, fundamentalmente, en ejercicios consistentes en la repetición de estructuras en las que se incluyen las unidades lingüísticas que se quieren aprender.

- *Método global-estructural.* La base principal de este método es el uso de los medios audiovisuales. Se centra demasiado en el lenguaje hablado descuidando otras habilidades como la lectura o la escritura.

- *Método cognitivo.* Derivado de la gramática generativa pretende que los alumnos puedan crear distintas manifestaciones gramaticales una vez hayan adquirido un conocimiento cognitivo del sistema lingüístico.

- *Método comunicativo.* El contexto en el que se produce la comunicación pasará ahora a tener un papel destacado en la enseñanza de la segunda lengua porque de ese contexto se derivarán una serie de aspectos lingüísticos en los que será necesario incidir para que, una vez aprendidos, se pueda participar en una comunicación real.

Jiménez, R. M. (1994) también analiza algunos métodos de enseñanza de un segundo lenguaje desde el punto de vista del aprendizaje del vocabulario y observa que el aprendizaje del vocabulario no ha tenido una explicación clara en ninguno de los cuatro métodos que ella menciona, advirtiendo, además, una ausencia total en los libros de texto de estrategias que se puedan poner en práctica a la hora de enseñanza y aprendizaje:

- El primer método del que se ocupa es el de Gramática y la Traducción. El aprendizaje de vocabulario era adquirido con listas de palabras con su correspondiente traducción.

- En el Método Directo se presenta directamente la nueva palabra en la lengua meta y los alumnos la relacionan con el objeto, acción, gestos, contexto, etc.

- El Método Audiolingual consiste en la audición de frases construidas y la repetición de estas a la vez que la creación de unas nuevas, logrando que los alumnos aprendan así la gramática y el vocabulario dentro de un contexto.

- Por último, durante la década de los ochenta podemos empezar a hablar de Método Comunicativo, el cual tampoco ofrece mejoría en la adquisición de nuevo vocabulario. Este método da mayor importancia a las funciones comunicativas que a las estructuras, dando por hecho que los alumnos aprenden las nuevas palabras de forma inconsciente y automática.

2.3. Metodología de software ágil para videojuegos:

2.3.1 SCRUM

SCRUM es uno de los más conocidos de los frameworks ágiles. Es la fuente de gran parte del pensamiento que se encuentra detrás de los valores del Manifiesto Ágil. En la Figura 5, se presenta los componentes del SCRUM:

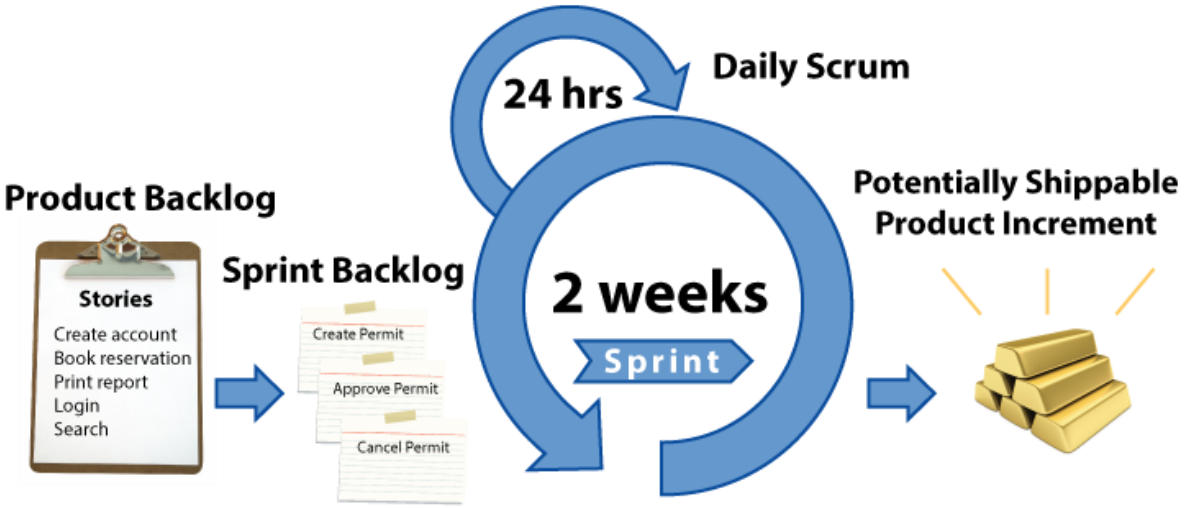


Figura 5: Descripción del SCRUM

Fuente: Alaimo (2013). *Proyectos ágiles con SCRUM*.

Según Alaimo, M. (2013, Scrum es un marco de trabajo que nos permite encontrar prácticas emergentes en dominios complejos, como la gestión de proyectos de innovación. No es un proceso completo, y mucho menos, una metodología.

Los valores del manifiesto ágil están muy interrelacionados con SCRUM:

- Individuos e interacciones entre los mismos sobre procesos y herramientas:
    - Todo framework ágil se basa directamente en la confianza puesta en los miembros de un equipo y la manera en la cual los mismos interactúan entre sí. Los equipos definen que hacer, como hacerlo y finalmente lo ejecutan. Para ello, estos detectan los obstáculos en su camino y asumen responsabilidad para resolver todos los problemas. Un detalle importante es que los equipos trabajan en conjunto con las demás partes de la organización para resolver asuntos o problemas fuera de su control, el cual es crucial para todo equipo. Si esto no llega a concretarse, habrá problemas más adelante en el equipo.
- Software funcionando sobre documentación extensiva:
    - SCRUM requiere un incremento constante del producto trabajado al final de cada sprint, lo que permite analizar, diseñar y realizar pruebas del progreso del proyecto.
- Colaboración con el cliente sobre la negociación contractual:
    - En SCRUM existe el Product Owner (o dueño del producto) el cual es el punto de contacto del equipo con los eventuales usuarios finales del producto. El Product Owner es parte del equipo de SCRUM y trabaja colaborativamente con los demás miembros, lo cual permite una mejor facilidad para brindar las tareas necesarias al grupo que debe realizar a continuación.
- Respuesta ante cambios sobre el seguimiento del plan:
    - En SCRUM, la metodología de trabajo está diseñada para que todos tengan la información necesaria para continuar el proyecto sin ningún contratiempo, con la toma de decisiones más efectiva debido a esto. Los avances del proyecto están representados por un incremento en el producto

real y su funcionamiento. Todo detalle del proyecto está disponible para todos los miembros del grupo mediante el backlog y los problemas que pueden surgir son discutidos con todo el grupo y resueltos de forma inmediata, lo cual es crucial, debido a que SCRUM funciona de manera correcta para equipos con capacidad de adaptarse a los problemas.

2.3.1.1 Valores de SCRUM:

Para trabajar bajo la metodología de SCRUM, se es necesario tener una base de valores específica. A través del trabajo en equipo y la mejora continua, SCRUM tanto creo como depende de estos valores (Figura 6):

- Foco:
    - Se hace enfoque a pocas cosas a la vez para trabajar mejor y producir un mejor resultado.
- Coraje:
    - Al no estar solos y teniendo apoyo de un equipo confiable, se desarrolla coraje para enfrentar desafíos más grandes.
- Apertura:
    - Aprender a manifestar las preocupaciones propias en relación al trabajo para tomarlas en cuenta.
- Compromiso:
    - Al tener mayor control del proyecto en sí, el grupo llega a comprometerse en el éxito del mismo.
- Respeto:
    - El trabajo en equipo hace que los mismos miembros se respeten los unos a los otros para el buen desarrollo y avance del proyecto.

Figura 6: Valores del SCRUM

Fuente: Dave West (2016) Recuperado de https://blog.scrum.org/updates-scrum-guide-5-scrum-values-take-center-stage/

2.3.1.2 El manifiesto ágil:

Según Alaimo, M. (2013), el manifiesto ágil está compuesto por 4 valores y 12 principios:

Valorar a las personas y las interacciones entre ellas por sobre los procesos y las herramientas

Las personas son el principal factor de éxito de un proyecto de software. Es más importante construir un buen equipo que construir el contexto. Muchas veces se comete el error de construir primero el entorno de trabajo y esperar que el equipo se adapte automáticamente. Por el contrario, la agilidad propone crear el equipo y que éste construya su propio entorno y procesos en base a sus necesidades.

Valorar el software funcionando por sobre la documentación detallada

La regla a seguir es "no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante". Estos documentos deben ser cortos y centrarse en lo esencial. La documentación (diseño, especificación técnica de un sistema) no es más que un resultado intermedio y su finalidad no es dar valor

en forma directa al usuario o cliente del proyecto. Medir avance en función de resultados intermedios se convierte en una simple "ilusión de progreso".

Valorar la colaboración con el cliente por sobre la negociación de contratos

Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta mutua colaboración será la que dicte la marcha del proyecto y asegure su éxito.

Valorar la respuesta a los cambios por sobre el seguimiento estricto de los planes:

La habilidad de responder a los cambios que puedan surgir a lo largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también su éxito o fracaso. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores anteriores son los pilares sobre los cuales se construyen los doce principios del Manifiesto Ágil. De estos doce principios, los dos primeros son generales y resumen gran parte del espíritu ágil del desarrollo de software, mientras que los siguientes son más específicos y orientados al proceso o al equipo de desarrollo:

- La mayor prioridad es satisfacer al cliente a través de entregas tempranas y frecuentes de software con valor.
- Aceptar el cambio incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan los cambios para darle al cliente ventajas competitivas.
- Entregar software funcionando en forma frecuente, desde un par de semanas a un par de meses, prefiriendo el periodo de tiempo más corto.
- Expertos del negocio y desarrolladores deben trabajar juntos diariamente durante la ejecución del proyecto.
- Construir proyectos en torno a personas motivadas, generándoles el ambiente necesario, atendiendo sus necesidades y confiando en que ellos van a poder hacer el trabajo.
- La manera más eficiente y efectiva de compartir la información dentro de un equipo de desarrollo es la conversación cara a cara.

- El software funcionando es la principal métrica de progreso.
- Los procesos ágiles promueven el desarrollo sostenible. Los sponsors, desarrolladores y usuarios deben poder mantener un ritmo constante indefinidamente.
- La atención continua a la excelencia técnica y buenos diseños incrementan la agilidad.
- La simplicidad –el arte de maximizar la cantidad de trabajo no hecho- es esencial.
- Las mejores arquitecturas, requerimientos y diseños emergen de equipos auto-organizados.
- A intervalos regulares, el equipo reflexiona acerca de cómo convertirse en más efectivos, luego mejora y ajusta su comportamiento adecuadamente.

2.3.1.3 Roles de SCRUM:

En la Figura 7, se presenta un resumen de los roles en un entorno de SCRUM:

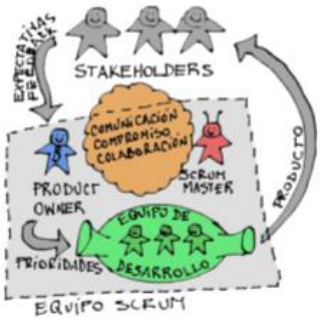


Figura 7: Roles en SCRUM

Fuente: Alaimo (2013). *Proyectos agiles con SCRUM*.

2.3.1.3.1 Dueño del producto (Product Owner):

Es la única persona responsable de delinear la fecha meta para el lanzamiento del producto. Este lo logra mediante la gestión del equipo, además de mantenimiento y refinamiento de los elementos en el Backlog.

2.3.1.3.2 Miembro del equipo de desarrollo:

El equipo de desarrollo está compuesto por profesionales que realizan el trabajo necesario para poder entregar el avance del producto. La metodología de SCRUM requiere que los miembros de este equipo sean de diferentes disciplinas, lo cual hace posible el reunir sus habilidades para la entrega del producto.

Los miembros del grupo tienen la responsabilidad de organizarse por su cuenta para lograr la meta del sprint establecido en su cronograma, procedimiento incrementalmente hasta el siguiente sprint.

2.3.1.3.3 SCRUM Master:

El SCRUM es el "líder servicial" que ayuda al resto del equipo a seguir el progreso definido por el sprint. Debe tener un interno conocimiento del método SCRUM para ser el indicado para el rol. Este también actúa como coach del equipo, ayudando los miembros a realizar sus roles en el equipo, además de asegurar que el SCRUM sea comprendido e implementado, tanto dentro como fuera del equipo.

Según Alaimo, M. (2013), las responsabilidades principales del SCRUM Master son:

- Velar por el correcto empleo y evolución de Scrum
- Facilitar el **uso de Scrum** a medida que avanza el tiempo. Esto incluye la responsabilidad de que todos asistan a tiempo a las daily meetings, reviews y retrospectivas.
- Asegurar que el equipo de desarrollo sea **multifuncional** y eficiente
- **Proteger** al equipo de desarrollo de distracciones y trabas externas al proyecto

- Detectar, monitorear y **facilitar la remoción de los impedimentos** que puedan surgir con respecto al proyecto y a la metodología
- Asegurar la **cooperación** y **comunicación** dentro del equipo.

2.3.1.4 Product Backlog:

El Product Backlog es la lista ordenada de ideas para el producto, mantenidas en el orden que el grupo está dispuesto a llevarlas a cabo, además de ser la fuente de información acerca de los requerimientos del mismo producto.

2.3.1.4.1 Actividades del Product Backlog:

2.3.1.4.1.1 Refinamiento:

Es una actividad constante a lo largo del proyecto de SCRUM, el cual consiste en:

- Mantener el backlog ordenado.
- Eliminar o degradar elementos del proyecto que ya no son importantes.
- Agregar y/o promover ítems que surgen o se vuelven importantes.
- Dividir ítems en ítems más pequeños.
- Unir ítems.
- Estimar ítems.

El beneficio más importante del refinamiento es que realiza la preparación de los subsiguientes sprints, para la agilización del proyecto en sí.

2.3.1.4.1.2 Planificación del Sprint:

Consiste en la cantidad de tiempo acotada para la realización del sprint. El equipo participa completamente en la planificación del sprint, con cada ítem del mismo siendo discutido entre todos para llegar al acuerdo del tiempo requerido para completar el objetivo del sprint. Debido a que el éxito de la planificación del sprint depende de la calidad del refinamiento del *backlog*, se recalca la importancia de la actividad de refinamiento.

La planificación del sprint está compuesta de dos partes:

- Determinación del trabajo a realizar en el sprint:
    - El Product Owner presenta los ítems del Product Backlog al equipo de desarrollo y de SCRUM completo para que todos tengan la idea del proyecto.
    - El equipo de desarrollo, para decidir cuantos ítems va a tomar en cuenta, toma en consideración el estado actual del incremento del producto, la performance del equipo en el pasado, la capacidad actual y el Product Backlog ordenado.
    - A menudo, aunque no siempre, se le da un objetivo al Sprint, llamado Objetivo del Sprint. Ésta es una práctica muy poderosa que ayuda a todos a enfocarse en la esencia de lo que debe ser realizado, minimizando la importancia de detalles que pueden no ser importantes.

- Determinación de cómo realizar el trabajo:
    - El equipo de desarrollo se reúne para decidir el siguiente incremento del producto. Se realiza el diseño y planificación mínima suficiente para poder completar el trabajo durante el sprint planeado.

- Decidir cómo hacer el trabajo es responsabilidad del Equipo de Desarrollo, así como decidir qué hay que hacer es responsabilidad del Product Owner.

2.3.1.5. Sprint Backlog:

Es la lista de ítems del Product Backlog, los cuales han sido refinados que han sido elegidos para el sprint actual, para poder realizar el trabajo en el tiempo ya decidido previamente.

Según Alaimo, M. (2013), el resultado de cada Sprint debe ser un incremento funcional potencialmente entregable:

Incremento funcional porque es una característica funcional nueva (o modificada) de un producto que está siendo construido de manera evolutiva. El producto crece con cada Sprint.

Potencialmente entregable: Porque cada una de estas características se encuentra lo suficientemente validada y verificada como para poder ser desplegada en producción (o entregada a usuarios finales) si así el negocio lo permite o el cliente lo desea.

2.3.1.5.1 Actividades del Sprint Backlog:

2.3.1.5.1.1 Definición de hecho (Done):

Al entregar un avance del proyecto, este tiene que estar definido como "hecho", en relación al término de hecho planteado por el grupo de SCRUM. A medida que el equipo madura, los requisitos a cumplir para realizar la condición de "hecho"

2.3.1.5.1.2 Scrum Diario:

El equipo de desarrollo se reúne para discutir el progreso del sprint. En esta reunión, se discuten ciertos temas en relación al proyecto:

- Lo logrado en el último Scrum diario.
- Lo que se piensa lograr en este Scrum.
- Lo que está impidiendo el avance del Scrum actual.

Se debe tener en mente que esta no es una reunión de reporte a gerencia. Es una comunicación entre los miembros del equipo de desarrollo para verificar que todo está en orden. A partir de lo discutido en la reunión, el equipo de desarrollo se reorganizará, si es necesario, para lograr el objetivo del Sprint.

2.3.1.5.1.3 Revisión del Sprint:

Al final de cada Sprint, las partes interesadas revisan el resultado del Sprint, con el punto central a tomar en cuenta es el incremento realizado en el desarrollo del producto durante el Sprint. En sí, la revisión del sprint es una reunión formal para determinar el avance realizado en el periodo definido por los grupos la cual revisa el estado actual del producto y realizar discusiones para mejorar el ritmo de desarrollo.

Los equipos encuentran sus propias formas de realizar la Revisión del Sprint. Es usual una demostración del Incremento de Producto. El grupo a menudo discute lo que observó durante el Sprint y qué ideas de producto le vinieron a la mente. Se discute el estado del Product Backlog y se habla acerca de posibles fechas de finalización y qué puede llegar a ser hecho para esas fechas.

2.3.1.5.1.4 Retrospectiva del Sprint:

Al final de cada Sprint, el Equipo Scrum se reúne para la Retrospectiva del Sprint. El propósito es revisar cómo fueron las cosas respecto al proceso, la relación entre las personas y las herramientas utilizadas. El equipo identifica qué salió bien y qué no tan bien, e identifica potenciales mejoras. Luego diseña un plan para mejorar las cosas a

futuro. Todas las reuniones en Scrum están acotadas por tiempo. La duración recomendada de la Retrospectiva del Sprint es una hora por cada semana de duración del Sprint.

2.4 Herramientas de desarrollo del software:

2.4.1 Unity 3D:

Unity 3D es una plataforma de desarrollo de no solo videojuegos con contenido 2D y 3D, sino para la generación de cortometrajes, a través de diferentes plataformas como PCs, Mac, Linux, iOS, Android, Windows Phone, Wii U, PS3 y Xbox360. En la Figuras 8 y 9 se muestran algunas estadísticas relacionadas con la herramienta de Unity.

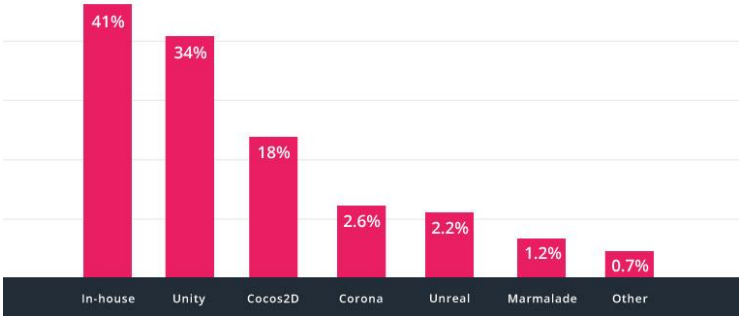


Figura 8: Escala de juegos móviles creados por diferentes consolas

Fuente: SourceDNA, Q1 2016

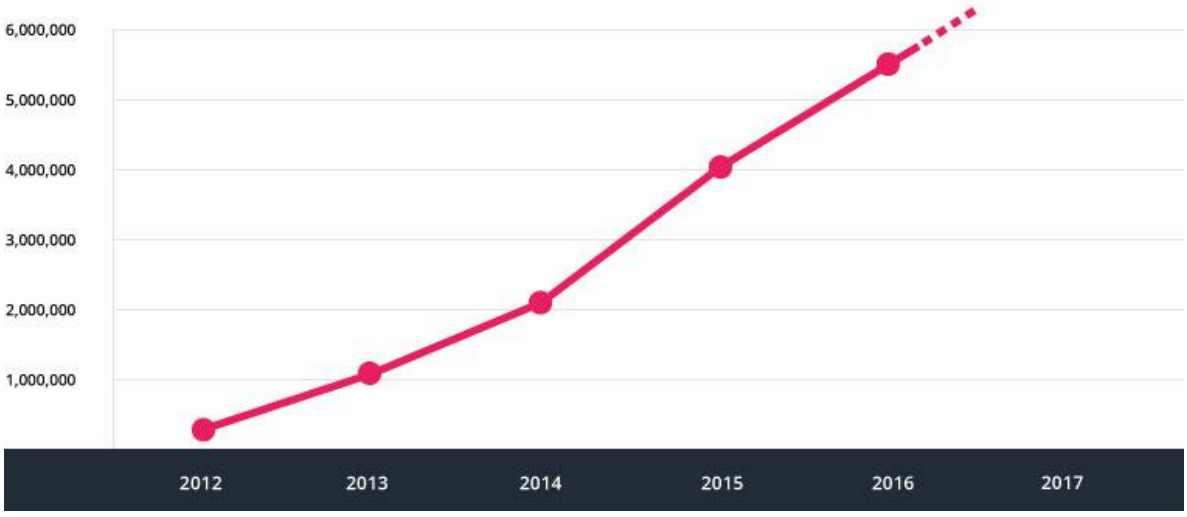


Figura 9: Grafica de la cantidad de usuarios registrados que usan Unity del 2010-2014

Fuente: Unity, 2016

Unity no se limita solo a la creación de videojuegos. Otros clientes que usan Unity3D son Coca-Cola, Disney, Electronic Arts, LEGO, Microsoft, la NASA, Nexon, Nickelodeon, Square Enix, Ubisoft, Obsidian, Insomniac y Warner Bros. Desde los estudios grandes y pequeños hasta los profesionales independientes, cada vez es mayor el número de desarrolladores que está pasándose a Unity.

2.4.1.1 C#:

C# (pronunciado *si sharp* en inglés) es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común orientada a objetos. (C#. *Wikipedia* (s.f.))

Unity3D utiliza el lenguaje de programación en base a C# para la creación de scripts o clases propias, dependiendo de la necesidad del programador. El proyecto de videojuego se trabajó completamente desde cero usando C#.

2.4.2 Base de datos:

Una **base de datos** o **banco de datos** es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. (Base de datos. *Wikipedia* (s.f.)). Es decir, se le llama base de datos a los bancos de información que contienen datos relativos a diversas temáticas y categorizados de distinta manera, pero que comparten entre sí algún tipo de vínculo o relación que busca ordenarlos y clasificarlos en conjunto dependiendo de la estructura para consultarlos en un tiempo futuro.

El tipo de información que se puede grabar en una base datos varía, desde simple cadenas de caracteres o fechas, imágenes o archivos de sonido hasta enteros libros o archivos de mayor peso. Dicha información es manipulada por el uso de sentencias (query) para crear tablas, ingresar o borrar información o seleccionar data ya ingresada para el

análisis o búsqueda deseada (CREATE, ALTER TABLE, SELECT, DELETE, entre otros).

2.4.2.1 SQLite:

SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña (~275 kiB)[2] biblioteca escrita en C. A diferencia de los sistema de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo (SQLite. *Wikipedia* (s.f.)). El código de SQLite es de acceso público, volviéndola de uso libre, sea para uso comercial o privado, lo cual lo vuelve una de las plataformas de generación de base de datos más distribuida por todo el mundo.

Los tipos de base de datos generados por SQLite son cross-plataform, reforzando su característica de ser transferibles a través de diferentes equipos. Aparte de abrir base de datos de tipo .sqlite, también puede abrir y manipular otras bases de datos con diferentes términos (como .db por ejemplo).

2.4.3 UX (Diseño de experiencia del usuario):

UX, o diseño de la experiencia del usuario, es el proceso de mejorar la experiencia y satisfacción de un usuario al usar un producto por medio de la mejora de la accesibilidad y usabilidad de la interacción del usuario con el producto. El diseño de experiencia del usuario involucra HCI (Human-Computer Interaccion) o interacción humano-computadora, para crear escenas de transición y menús que fueran fáciles de entender y visualizar por los posibles usuarios a futuro. (User Experience Design. Wikipedia (s.f.))

Por ejemplo, algunos ejemplos populares de la efectividad de UX en general se pueden apreciar en la composición de algunas páginas de compra por Internet o páginas de reservación de hoteles en sí. La teoría de UX sostiene que la interfaz con la cual el usuario interactúa debe ser fácil de entender para que el usuario pueda conseguir o comprar su objetivo deseado sin ninguna dificultad o malestar.

2.5 La memoria a largo plazo:

Según Sprenger, M. (2005), existen ciertos factores relacionados en el aprendizaje, que pueden llevar a aprender y comprender nuevos conceptos con facilidad:

- Frecuencia: En la lectura, se ha determinado que, si una persona lee constantemente, este mejorara en actividades lectoras. Con repetida exposición al tema que se desea aprender, el usuario o estudiante aprenderá con mejor facilidad dicho tema.

- Intensidad: Aprender requiere practica rigorosa. Un estudiante lograra aprender habilidades relacionadas a su aprendizaje en un corto periodo de tiempo si es que este logra practicar intensamente.

- Entrenamiento cruzado: Enseñanza para la memoria requiere fuertes conexiones neuronales con otras conexiones. Por ello, diferentes habilidades con diferentes formas de memoria deben de ser usadas.

- Adaptabilidad: Para enseñar a retener la memoria se requiere que el profesor este constantemente monitoreando el progreso del estudiante y ajustar la situación de enseñanza/aprendizaje para que se adapte a las necesidades del mismo.

- Motivación y atención: Estos son los factores que mantienen al estudiante interesado en el proceso de aprendizaje. Varias estrategias y técnicas son necesarias para lograr mantener la atención del estudiante en el aprendizaje.

Además, según el mismo autor, existen siete pasos para el aprendizaje de la memoria a largo plazo (Figura 10) y la Tabla 2 muestra las características de cada paso:

- *Reach* (alcanzar): Para lograr alcanzar a retener información en la memoria a largo plazo, no se tiene que obtener habilidades de motivación o atención. Lo único necesario es poner un poco de novedad en los procesos de aprendizaje. Poner emoción o dar algún tipo de incentivo ayuda a generar un hábito de aprendizaje.

- *Reflect* (reflexión): La reflexión de lo aprendido permite a los estudiantes a buscar en la memoria acerca de información por algún fragmento de conocimiento que puedan tener acerca del tema que están aprendiendo. Al manipular la nueva información obtenida con la información retenida se conectará la nueva información con más duraderas memorias de largo plazo.

- *Recode* (reestructurar): Mientras que la información se encuentra en la memoria en proceso, los estudiantes pueden definir sus conceptos propios para la nueva información. Si los estudiantes logran definir su propia explicación para el contexto/información, esta se retendrá en la memoria de largo plazo.

- *Reinforce* (reforzar): En este paso, se pregunta a los alumnos si es que comprenden las ideas, conceptos y procedimientos de lo aprendido. Al revisar el *feedback* de los alumnos sin calificarlo, se pueden aclarar algunas dudas que puedan tener los alumnos.

- *Rehearse* (practicar): Cuando los alumnos han logrado determinar sus propios conceptos y significados para los temas que quisieran aprender, es momento de pasar la información a la memoria a largo plazo, por medio del uso de diferentes técnicas, como el ensayo general de las ideas. Para poder grabar la información en la memoria de largo plazo, múltiples ensayos son necesarios. Si estos no son practicados con constancia, la información no va a ser correctamente recordada en la memoria de largo plazo.

- *Review* (revisar): Mientras la fase de practicar se enfoca en colocar la información en la memoria de largo plazo, la fase de revisar busca extraer y manipular la información grabada en la memoria en trabajo. Los productos de la manipulación luego pueden ser movidas a la memoria de largo plazo.

- *Retrieve* (extraer): La habilidad de extraer las memorias de largo plazo, llevándolos a la memoria de trabajo actual, y resolver problemas en el contexto del alumno. La extracción de la información resulta ser más fácil cuando el contexto del problema a resolver resulta ser similar al contexto en el cual se ha realizado el aprendizaje.

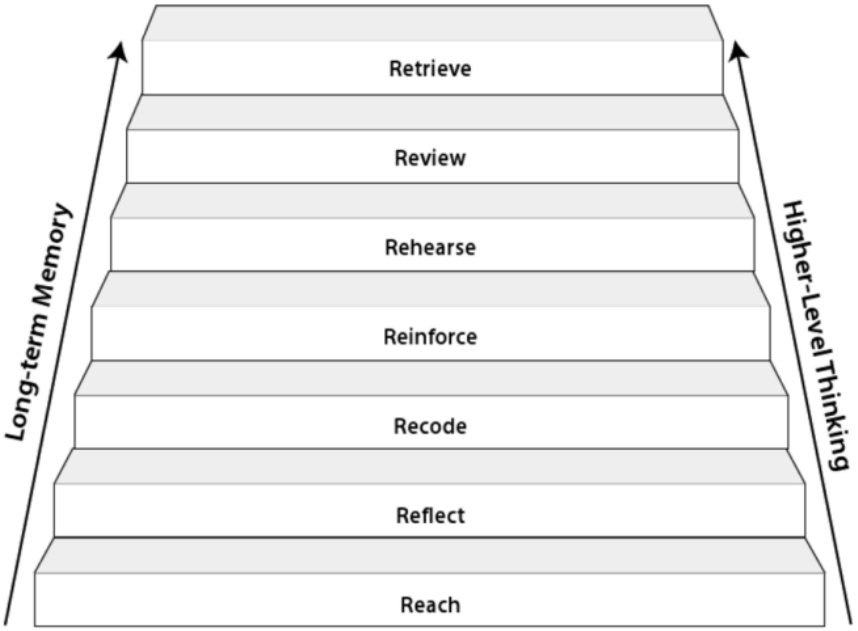


Figura 10: Los siete pasos para el aprendizaje de largo plazo.

Fuente: Sprenger, M. (2005) *How to Teach so Students Remember*

PASOS	CARACTERISTICAS	PROCESO DE MEMORIA
ALCANCE	Atencion, Motivacion, Emocion, Estilos de Aprendizaje	Sensorial > Inmediato
REFLECCION	Preguntas, Colaborar, Visualizar	Inmediato > Memoria en trabajo constante
RECODIFICAR	Autogenerar, Simbolizar, Dialogo	Memoria en trabajo constante
REFORZAR	*Feedback*, Reenseñanza, Reinvención	Memoria en trabajo constante, Emocional
PRACTICA	Repetir, Memorizacion por repeticion, Elaborar	Memoria en trabajo constante > Memorizaje a largo plazo
REVISION	Instrucción equivalente, Anticipación de problemas	Memoria a largo plazo > Memoria en trabajo constante > Memoria a largo plazo
RECIBIR	Interiorizacion de pistas e información, Prueba de ansiedad	Memoria a largo plazo > Memoria en trabajo, Memoria emocional

Tabla 2: Tabla de los siete pasos para la memoria a largo plazo

Fuente: Sprenger, M. (2005) *How to Teach so Students Remember*

2.5.1. Memrise

Según el artículo publicado Walker, L. (2015), *Memrise* es una plataforma online de apoyo para el auto estudio que el autor utilizo para aprender vocabulario en latin. Sus mayores beneficios eran su portabilidad, accesibilidad (via smartphones o computadoras) y facilidad de uso. Fue creado por investigadores en psicología y lingüística, con el propio de apoyar y enriquecer el desarrollo de la memoria a largo plazo en relación al vocabulario.

Memrise utiliza un sistema de revisión de algoritmo que fuerza a los estudiantes a revisitar lecciones (o mundos, dentro del contexto del programa) repetidamente, en un tiempo particular programado por la misma plataforma, con recordatorios ocurriendo en ocasiones en las cuales la memoria se muestra más propensa a olvidar. Mayor enfoque se brinda a palabras los cuales los estudiantes frecuentemente olvidan, motivando un esfuerzo más concentrado en las partes más débiles de la memoria, en especial la memoria a largo plazo.

2.6 IA (Inteligencia Artificial):

Según Russell, J. &, Norvig, P. (2004), la definición de inteligencia artificial presenta diferentes permutaciones (Tabla 3). Las definiciones de la izquierda miden el éxito en términos de la fidelidad en la forma de actuar de los humanos mientras que las de la derecha toman como referencia un concepto ideal de inteligencia, que llamaremos racionalidad. Un sistema es racional si hace «lo correcto», en función de su conocimiento.

Sistema que piensan como humanos	**Sistemas que piensan racionalmente**
"El nuevo y excitante esfuerzo de hacer que los computadores piensan…máquinas con mentes, en el más amplio sentido literal" (Haugekand, 1985). "La automatización de actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje…" (Bellman,1978).	"El estudio de las facultades mentales mediante el uso de modelos computacionales" (Charniak y McDermott, 1985). "El estudio de los cálculos que hacen posible percibir, razonar y actuar" (Winston,1992).
Sistemas que actúan como humanos	**Sistema que actúan racionalmente**
"El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizados por personas requieren de inteligencia" (Kurzwell,1990). "El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor" (Rich y Knight, 1991).	"La Inteligencia Computacional es el estudio del diseño de agente inteligentes" (Poole *et al*., 1998). "IA…está relacionada con conductas inteligentes en artefactos" (Nilssonm.1998).

Tabla 3: Tabla de significados de inteligencia artificial.

Fuente: Russell, J. &, Norvig, P. (2004), *Inteligencia artificial. Un Enfoque Moderno*.

En sí, la inteligencia artificial resulta en emular el comportamiento racional humano para la toma de decisiones en diversos contextos que requieran una solución. Como define Alan Turing (1950) en su propuesta llamada **la prueba de Turing** para determinar una definición operacional y satisfactoria de inteligencia.

Hoy por hoy, podemos decir que programar un computador para que supere la prueba requiere un trabajo considerable. El computador debería poseer las siguientes capacidades:

- Procesamiento del lenguaje natural que le permita comunicarse satisfactoriamente en inglés.

- Representación del conocimiento para almacenar lo que se conoce o siente.
- Razonamiento automático para utilizar la información almacenada para responder a preguntas y extraer nuevas conclusiones.
- Aprendizaje automático para adaptarse a nuevas circunstancias y para detectar y extrapolar patrones.
- Visión computacional para percibir objetos.
- Robótica para manipular y mover objetos.

Estas seis disciplinas abarcan la mayor parte de la IA. Los investigadores del campo de la IA han dedicado poco esfuerzo a la evaluación de sus sistemas con la Prueba de Turing, por creer que es más importante el estudio de los principios en los que se basa la inteligencia que duplicar un ejemplar.

2.6.1 Agentes inteligentes:

Según Russell, J. &, Norvig, P. (2004), un agente es cualquier cosa capaz de percibir su medioambiente con la ayuda de sensores y actuar en ese medio utilizando actuadores. La Figura 11 ilustra esta idea simple. Un agente humano tiene ojos, oídos y otros órganos sensoriales además de manos, piernas, boca y otras partes del cuerpo para actuar. Un agente robot recibe pulsaciones del teclado, archivos de información y paquetes vía red a modo de entradas sensoriales y actúa sobre el medio con mensajes en el monitor, escribiendo ficheros y enviando paquetes por la red.

Para dar como ejemplo el comportamiento entre un agente y lo que detecta del mundo a su alrededor por medio de sensores, se tomara el ejemplo de una aspiradora extraída del mismo libro. Este mundo en particular tiene solamente dos localizaciones: cuadrícula A y B. La aspiradora puede percibir en qué cuadrante se encuentra y si hay suciedad en él. Puede elegir si se mueve hacia la izquierda, derecha, aspirar la suciedad o no hacer nada. Una función muy simple para el agente vendría dada por: si la cuadrícula en la que se encuentra está sucia, entonces aspirar, de otra forma cambiar de cuadrícula. Una muestra parcial de la función del agente representado en forma de tabla parcial de la función de agente sencilla para la aspiradora.

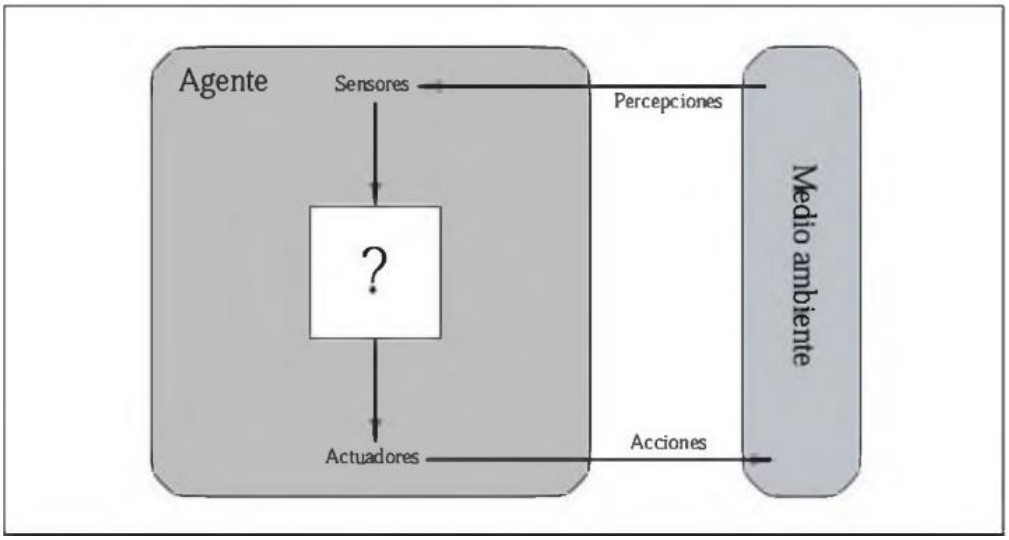


Figura 11. Comportamiento de un agente.

Fuente: Russell, J. &, Norvig, P. (2004), *Inteligencia artificial. Un Enfoque Moderno*.

Si un sistema logra cambiar entre estados sin ningún input externo y solo mediante la información recopilada por sus sensores, se puede argumentar que el sistema o programa se comporta de manera racional. (Tabla 4)

Secuencia de percepciones	Acción
[A. Limpio]	Derecha
[A. Sucio]	Aspirar
[B. Limpio]	Izquierda
[B, Sucio]	Aspirar
[A, Limpio], [A, Limpio]	Derecha
[A, Limpio], [A, Sucio]	Aspirar
-	-
-	-
-	-
[A, Limpio], [A, Limpio], [A,Limpio]	Derecha
[A, Limpio], [A, Limpio], [A, Sucio]	Aspirar

Tabla 4: Tabla parcial de una función de agente sencilla para el mundo de la aspiradora

Fuente: Russell, J. &, Norvig, P. (2004), *Inteligencia artificial. Un Enfoque Moderno*.

2.6.2. El entorno de trabajo:

Los entornos de trabajo son esencialmente el conjunto de problemas que el agente inteligente deberá resolver. Según Russell, J. &, Norvig, P. (2004), cada entorno de trabajo consiste medidas de rendimiento, el entorno en sí y los actuadores y sensores del agente, usando el acrónimo REAS para un entendimiento más resumido (Rendimiento, Entorno, Actuadores, Sensores). Como ejemplo, en la Tabla 5, se muestra un ejemplo de entorno de trabajo considerando como agente a un taxista.

Tipo de agente	Medidas de rendimiento	Entorno	Actuadores	Sensores
Taxista	Seguro, rápido, legal, viaje, confortable, maximización del beneficio.	Carreteras, otro tráfico, peatones, dientes.	Dirección, acelerador, freno, señal, bocina, visualizador.	Cámaras, sónar, velocímetro, GPS, tacómetro, visualizador de la aceleración, sensores del motor, teclado.

Tabla 5: Tabla de ejemplo de REAS de un taxista.

Fuente: Russell, J. &, Norvig, P. (2004), *Inteligencia artificial. Un Enfoque Moderno*.

2.6.2.1 Propiedades del entorno de trabajo:

El rango que pueden tomar los entornos de trabajo de los agentes inteligentes fluctúa en variedad. Sin embargo, se puede identificar un pequeño número de dimensiones en las que categorizar estos entornos. Estas dimensiones determinan, hasta cierto punto, el diseño más adecuado para el agente y la utilización de cada una de las familias principales de técnicas en la implementación del agente (Figura 12):

- Totalmente observable: Si los sensores del agente le proporcionan acceso al estado completo del medio en cada momento, entonces se dice que el entorno de trabajo es totalmente observable.

- Parcialmente observable: Un entorno puede ser parcialmente observable debido al ruido y a la existencia de sensores poco exactos o porque los sensores no reciben información de parte del sistema,

- Determinista vs Estocástico: Si el siguiente estado del medio está totalmente determinado por el estado actual y la acción ejecutada por el agente, entonces se dice que el entorno es determinista; de otra forma es estocástico.

- Episódico: En un entorno de trabajo episódico, la experiencia del agente se divide en episodios atómicos. Cada episodio consiste en la percepción del agente y la realización de una única acción posterior.

- Secuencial: En entornos secuenciales, por otro lado, la decisión presente puede afectar a decisiones futuras.

- Estático vs Dinámico: Si el entorno puede cambiar cuando el agente está deliberando, entonces se dice que el entorno es dinámico para el agente; de otra forma se dice que es estático.

- Discreto vs Continuo: La distinción entre discreto y continuo se puede aplicar al estado del medio, a la forma en la que se maneja el tiempo y a las percepciones y acciones del agente.

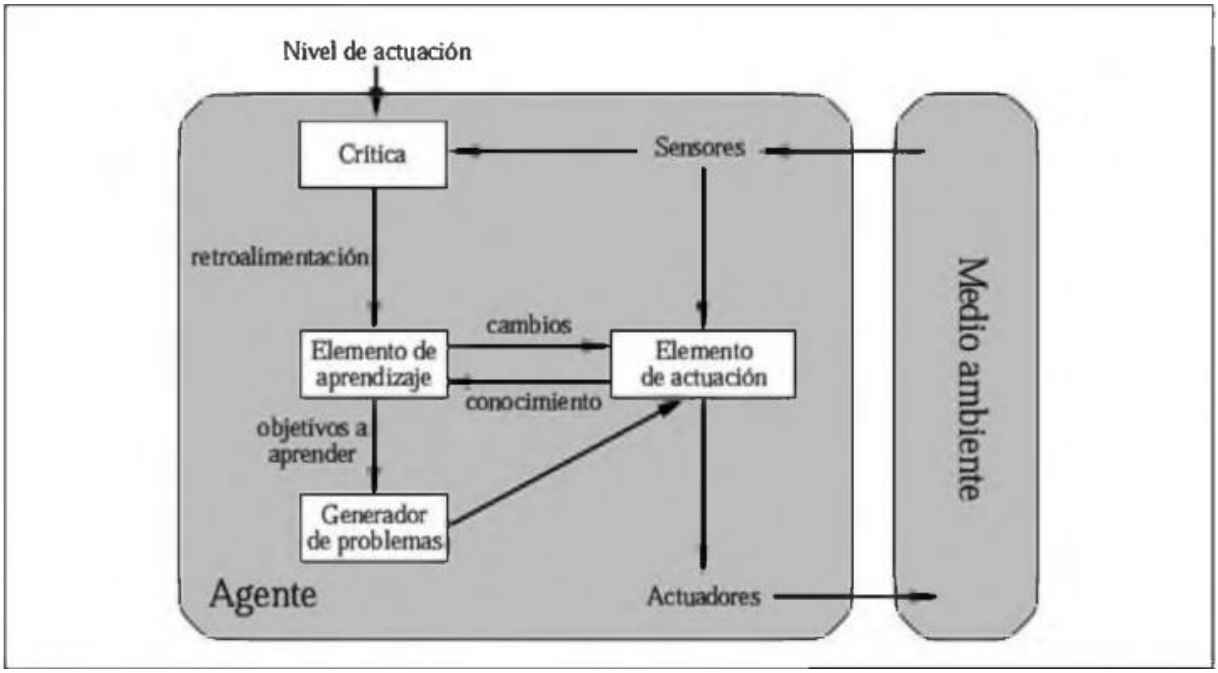


Figura 12: Modelo genérico de los agentes que aprenden.

Fuente: Russell, J. &, Norvig, P. (2004), *Inteligencia artificial. Un Enfoque Moderno*.

2.6.3 Agentes que aprenden:

Un agente que aprende se puede dividir en cuatro componentes conceptuales, tal y como se muestra en la Figura 12. La distinción más importante entre el elemento de aprendizaje y el elemento de actuación es que el primero está responsabilizado de hacer mejoras y el segundo se responsabiliza de la selección de acciones externas. El elemento de actuación es lo que anteriormente se había considerado como el agente completo: recibe estímulos y determina las acciones a realizar. El elemento de aprendizaje se realimenta con las críticas sobre la actuación del agente y determina cómo se debe modificar el elemento de actuación para proporcionar mejores resultados en el futuro.

2.7 Redes neuronales:

Las redes neuronales, o redes neuronales artificiales, para ser más precisos, representan a una tecnología que se origina desde varias disciplinas, como neurociencias, matemáticas, estadísticas, etc. En su forma más general, una red neuronal es una máquina que está diseñada para simular la manera como la cual un cerebro humano resuelve un problema en particular (Haykin, S. 2009). Los aspectos que las redes neuronales toman para asemejarse al cerebro son:

- Conocimiento es adquirido del ambiente por la red a través de un proceso de aprendizaje.
- Conexiones inter-neuronales, con sus respectivos pesos, son usadas para almacenar el conocimiento adquirido, cuyo proceso es conocido como el algoritmo de aprendizaje, cuya función involucra modificar los pesos sinápticos de la red para obtener el objetivo esperado.

Otro ejemplo que expresa la funcionalidad de la red neuronal es la forma en la cual un murciélago procesa la información obtenida de un sonar. El murciélago logra procesar información acerca de la distancia, tamaño y altura de su alimento, con un cerebro diminuto. Sin embargo, este cerebro aún puede procesar la señal sonora y extraer la información necesaria para poder encontrar su presa. La red neuronal tiene la capacidad de aprender y crecer, permitiendo generar sus propias reglas y realizar operaciones complejas computacionales.

La forma en la cual estas redes logran tener buen rendimiento es debido a que las redes están compuestas por diversas células simples, referidas como “neuronas” o unidades procesadoras, lo cual permite a la red “aprender” por medio de experiencias de su entorno. El nombre común para la denominación de este proceso es conocido como el algoritmo de aprendizaje, cuyo procedimiento involucra la modificación de los pesos sinápticos de la red, para así lograr obtener el resultado deseado.

2.7.1 El cerebro humano:

El sistema nervioso humano puede ser visto como un sistema de tres fases, como mostrado en la Figura 13 (Haykin, S., 2009).

Figura 13: Representación en bloques de un sistema nervioso

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

El cerebro está representado por una red neuronal que continuamente está recibiendo información la cual es procesada para la toma de decisiones. Como muestra dicha ficha, la información puede fluir de izquierda a derecha indican una transmisión de información en *forward* y las señales con flechas azules indican un flujo de *feedback* en el sistema. Un ejemplo simple para entender este comportamiento es el flujo de información entre el cerebro y los receptores de la piel cuando estos detectan presión o estimulo. El cerebro envía señales a través de sus neuronas a los receptores de la piel, los cuales procesan el estímulo externo y lo devuelven como información para que el cerebro pueda procesar sus decisiones en base a ello.

Otro concepto importante del cerebro es acerca de su plasticidad, es decir, su flexibilidad para adaptarse a los cambios de su entorno. Esto se traduce en la capacidad de creación de nuevas conexiones sinápticas entre neuronas y la modificación de neuronas existentes. En el cerebro, existen diferentes categorías de organización anatómica de baja a larga escala (Figura 14).

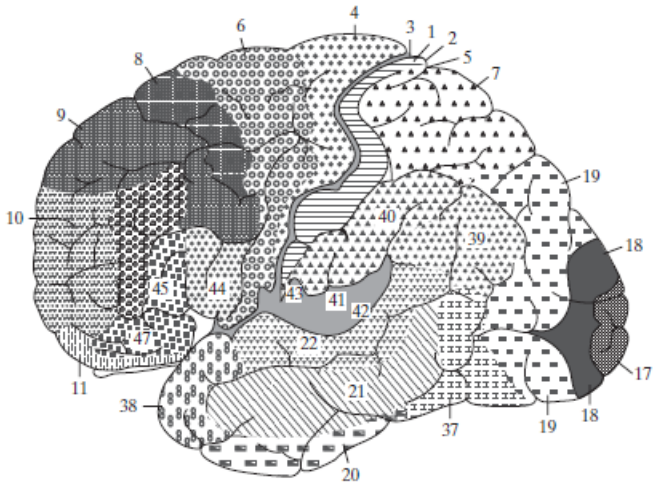


Figura 14: Regiones del cerebro y sus áreas

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

Las sinapsis representan el nivel más fundamental de transmisión de información, dependiendo de moléculas e iones para su acción. Los mapas topográficos se organizan para responder al ingreso de información sensorial, los cuales están comúnmente ordenados en categorías, donde las áreas visuales, auditoras y somatosensoriales (*soma*, cuerpo*; sensus*, sentido. Implica lo relativo a lo que pertenece a la sensibilidad del cuerpo) casi siempre juntas de manera que las áreas del cerebro que reaccionan al estímulo estén una cerca a la otra.

La Figura 15, muestra un mapa cito-arquitectural de la corteza cerebral. Esta muestra que diferentes ingresos sensoriales (motor, somatosensorial visual, etc.) están ubicados en áreas del cerebro de manera ordenada.

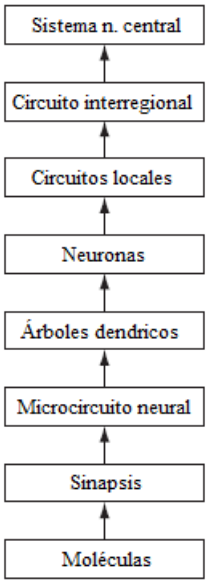


Figura 15: Organización estructural en el cerebro

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

2.7.2 Modelos de una neurona:

Una neurona es una unidad procesadora de información que es fundamental para la operación de una red neuronal. Estas poseen tres elementos básicos:

- Set de sinapsis o *links*: Cada una de ellas posee un *peso* propio. Las transmisiones de información entre las neuronas están multiplicadas por el peso de la conexión.
- Sumador: Suma las señales de entrada, controlado por el peso de sus respectivas neuronas.
- Función de activación: Limita la amplitud de la salida de la neurona. Esta también se puede referir como una función *reguladora* de las salidas.

Adicionalmente, según se muestra en la Figura 16 en el modelo neuronal se aplica una valor externo conocido como *bias*, denotado como b_k. El efecto del *bias* consiste en aumentar o reducir el ingreso neto de la función de activación, dependiendo si este es positivo o negativo.

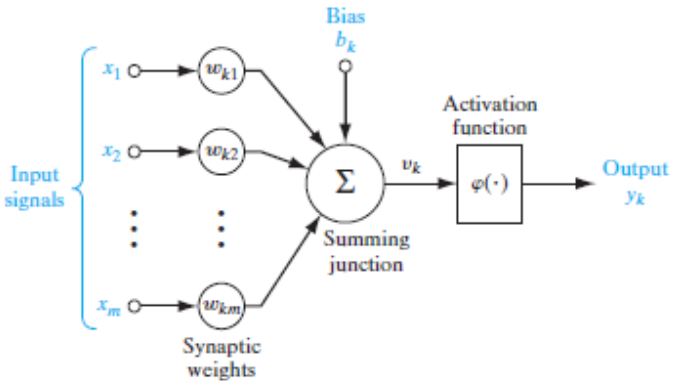


Figura 16: Modelo no linear de una neurona k

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

En términos matemáticos, según Haykin, S. (2009), se puede describir la neurona *k* como:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \quad \text{y} \quad y_k = \varphi(u_k + b_k) \qquad Ecuación\ (1)$$

Donde $x_1, x_2 \dots x_m$ son las señales de entrada; $w_{k1}, w_{k2} \dots w_{km}$ son los pesos de la neurona *k*; u_k (el cual no se muestra) es la *salida combinada linear* de todas las señales; b_k es el bias; $\varphi(.)$ es la función de activación y y_k la señal de salida de la neurona.

El uso del bias tiene un efecto de aplicar una transformación afinadora en la salida u_k:

$$v_k = u_k + b_k \qquad Ecuación\ (2)$$

Equivalentemente, usado las anteriores formulas, se puede redefinir la fórmula del bias a:

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \qquad Ecuación\ (3)$$

Dependiendo del *bias*, si este es positivo o no, se puede reformular el modelo de la neurona *k*, como se muestra en la Figura 17. Se puede resumir que el *bias* puede tener dos efectos: añadir una nueva señal de entrada y añadir un nuevo peso sináptico igual al *bias* b_k. Aunque las apariencias de ambas Figuras 16 y 17 son diferentes, matemáticamente son ambas iguales.

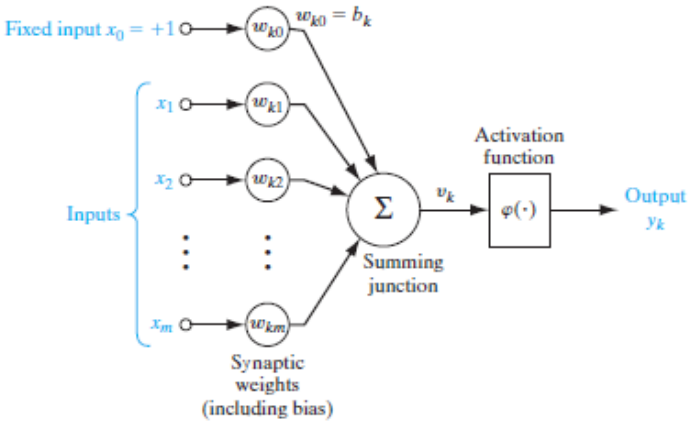


Figura 17: Otro modelo no linear, agregando el *bias* como entrada

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

2.7.2.1 Funciones de activación:

La función de activación, denotado como $\varphi(v)$, define el *output* de una neurona en términos de un campo local inducido *v*. (Haykin, S., 2009).

Existen tres tipos básicos de funciones de activación:

- Función *threshold* (Heaviside): Se puede entender como el estado positivo o negativo que puede tomar la función dependiendo de cuál sea el output de la neurona, y solo puede ser 1 para casos positivos ó 0 para el resto de casos (Figura 18).

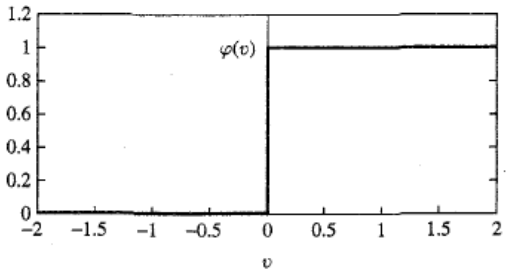


Figura 18: Función de threshold

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

- Función lineal: En geometría y álgebra elemental, una función lineal es una función polinómica de primer grado; es decir, una función cuya representación en el plano cartesiano es una línea recta. (Figura 19).

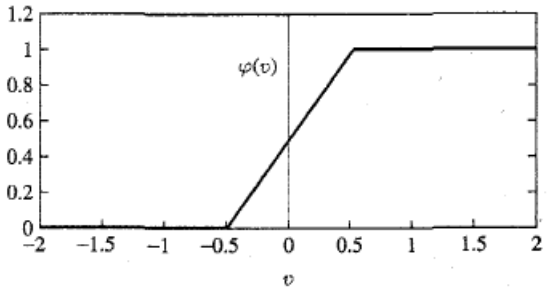


Figura 19: Función lineal

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

- Función Sigmoide: Esta función es la forma más común de función de activación utilizada para la construcción de redes neuronales (Figura 20). Su característica curvada se debe al resultado de una función logística como:

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \qquad Ecuación\ (4)$$

En el cual *a* es el grado de curvatura de la función.

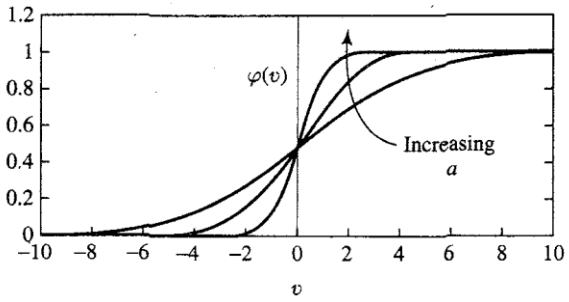


Figura 20: Función Sigmoide

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

Para el caso práctico de esta tesis, se utilizo la función de activación **sigmoidal**, al ser la función de activación más común usada en la construcción de redes neuronales artificiales.

2.7.3 Redes neuronales como gráficos dirigidos:

Un gráfico de un solo sentido es una red de *links* (ramas) que están interconectados en ciertos puntos llamados *nodos*. Las señales fluyen por dichos nodos siguiendo tres reglas básicas:

1. Una señal fluye a través de un *link* en la dirección definida por la flecha en el link:
    - Existen dos tipos de *links*:
        - Links sinápticos: Gobernado por una relación entrada-salida *linear* (Figura 21.a) cuya señal es multiplicada por su peso.
        - Links de activación: Gobernado por una relación entrada-salida *no linear*. (Figura 21.b).

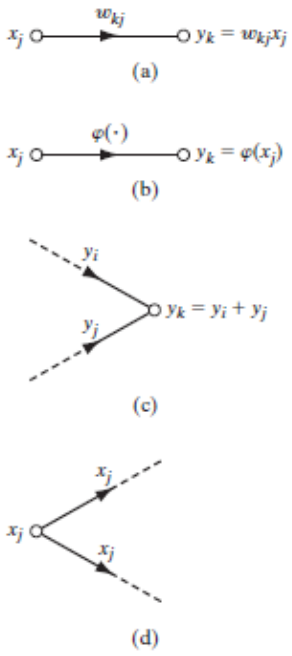


Figura 21: Ilustración gráfica de las reglas básicas para la construcción de flujos

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

2. Una señal de nodo es igual a la suma algebraica de todas las señales entrando al nodo pertinente por medio de sus *links*. (Figura 21.c)

3. La señal en un nodo es transmitida a todos sus links de salida siendo la transmisión enteramente independiente de las funciones de transferencia de estos *links*. (Figura 21.d).

   La red neuronal tiene la siguiente definición matemática al considerar las reglas previas:

- Una red neuronal es un gráfico dirigido consistiendo en nodos con interconexiones sinápticas y vínculos de activación, y es caracterizada por cuatro propiedades:
  - Cada neurona es representada por un set de vínculos lineales, un bias aplicado externamente y un posible vínculo de activación no linear.
  - Los vínculos sinápticos de una neurona regulan sus propias señales de entrada.
  - La suma total (con pesos) de las señales de entrada define el campo local de la neurona en cuestión.
  - El vínculo de activación regula la salida de la neurona.

2.7.4 Feedback:

El feedback existe en un sistema dinámico cuando el resultado de un elemento en el sistema influencia en parte a la entrada de dicho elemento, generando más puntos cerrados para la transmisión de señales a través de la red. La Figura 22 muestra un gráfico de flujo de un solo *loop*, donde $x_j(n)$ es la señal de ingreso, $x_j'(n)$ es la señal interna y $y_k(n)$, la señal de salida. El sistema se asume ser *linear*, consistiendo en un camino en forward y backward, caracterizado por los operadores A y B.

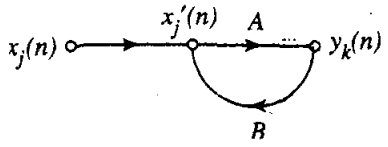


Figura 22: Ejemplo de un sistema singular recurrente

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

Matemáticamente, se expresa el gráfico como:

$$y_k(n) = A[x_j'(n)] \text{ y } x_j'(n) = x_j(n) + B[y_k(n)] \qquad \text{Ecuación (5)}$$

Eliminando x_j', nos quedamos con:

$$y_k(n) = \frac{A}{1 - AB}[x_j(n)] \qquad \textit{Ecuación (6)}$$

Se refiere a A/(1-AB) al operador del loop cerrado y a AB como el operador de loop abierto. Si consideramos que A es un peso fijo *w* y B es un operador con demora por unidad z^{-1}. La fórmula, entonces, cambiaria a:

$$\frac{A}{1 - AB} = \frac{w}{1 - wz^{-1}} \qquad \textit{Ecuación (7)}$$

Usando expansión binomial, se puede reescribir el código del operador cerrado como:

$$\frac{\mathrm{A}}{1-\mathrm{AB}} = w \sum_{l=0}^{inf} w^{l} z^{-l}\left[x_j(n)\right] \qquad Ecuación\ (8)$$

En relación a z^{-1}, tenemos:

$$z^{-1}\left[x_j(n)\right] = x_j(n-1) \qquad Ecuación\ (9)$$

Y con la información presente, se puede redefinir a la señal de salida $y_k(n)$ como la suma infinita pesada de las muestras pasadas y presentes de la señal de entrada $x_j(n)$:

$$y_k(n) = \sum_{l=0}^{inf} w^{l+1} x_j(n-l) \qquad Ecuación\ (10)$$

El comportamiento dinámico del sistema retroalimentativo se muestra en la Figura 23, controlado por el peso *w*. Podemos distinguir dos casos:

1. |w|<1, en el cual la señal de salida es exponencialmente convergente, es decir, el sistema es estable (Fig.23a)

2. |w|>=1, en el cual la señal es inestable. Si w = 1 entonces es una divergencia lineal; mayor a 1 resulta en una divergencia exponencial (Fig. 23c)

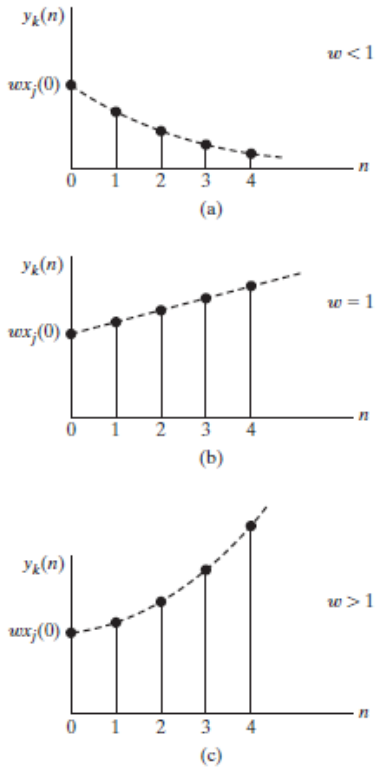


Figura 23: Tiempo de respuesta de la red para diferentes pesos *w*

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

2.7.5 Arquitectura de redes:

La manera en la cual las neuronas de la red neuronal están estructuras está íntimamente vinculada con el algoritmo de aprendizaje usado para entrenar la red. Es decir, los algoritmos de aprendizaje (reglas) usados están *estructurados.*

Las redes neuronales presentan las siguientes estructuras:

- Redes de una sola capa (Figura 24):
    - Las neuronas están divididas en capas, y la forma más simple de red posee una capa para los nodos de entrada y una capa de salida, con un solo sentido en *forward*.

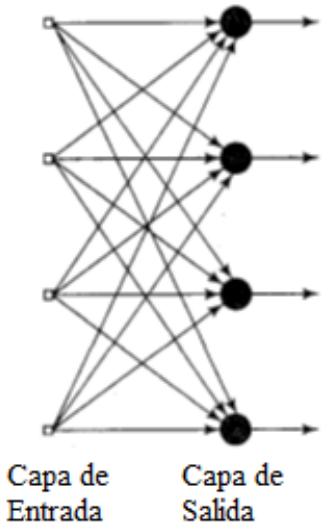


Figura 24: Ejemplo de una red de una sola capa

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

- Redes multicapa (Figura 25):
    - Las redes multicapa se caracterizan por poseer una o más capas ocultas, las cuales contienen neuronas escondidas que intervienen entre las capas de entrada y salida de la red. La función de estas neuronas es de intervenir entre el ingreso externo y la salida de la información de la red. Al añadir más capas, la red es capaz de extraer estadísticas de alto orden en relación a su ingreso. Se puede incluso fundamentar que la red gana una perspectiva global, sin importar de su conectividad local, debido al conjunto adicional de conexiones sinápticas y la dimensión adicional de interacciones neuronales.

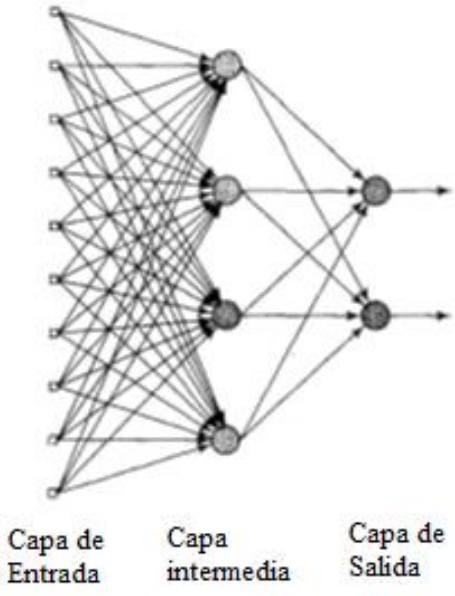


Figura 25. Ejemplo de una red multicapa capa

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

- Redes recurrentes:
    - Una red neuronal recurrente se distingue de las demás redes debido a que posee por lo menos un bucle lógico. En la Figura 26, se demuestra un ejemplo de una red recurrente que no posee bucles autorreferenciales, los cuales se caracterizan porque, las salidas de sus neuronas son regresadas a su misma neurona como entrada.
    - En la figura 27, se muestra otro ejemplo de red recurrente con neuronas escondidas, cuyas conexiones formadas se originan desde las neuronas ocultas al igual que las neuronas de salida. La presencia de bucles tiene un profundo impacto en la capacidad de aprendizaje de la red y en su *performance*.

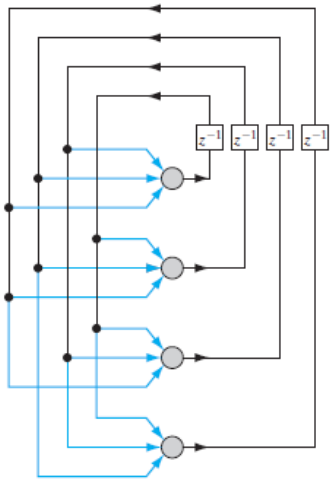


Figura 26: Ejemplo de una red recurrente no autorreferencial

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

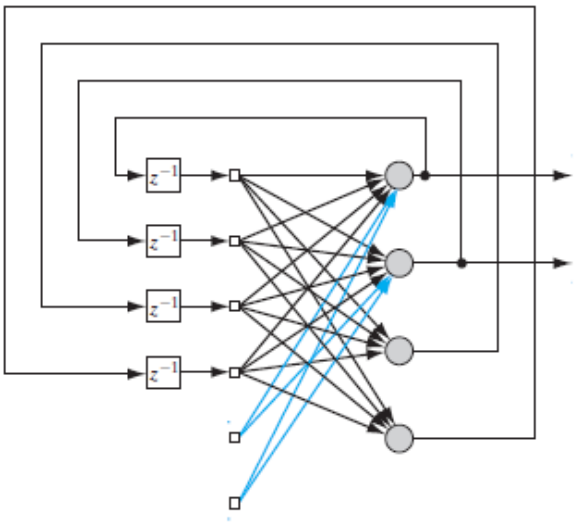


Figura 27: Ejemplo de red recurrente con una capa escondida

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

2.7.6 Representación del conocimiento:

El conocimiento se refiere a la información o modelos utilizados por una persona o máquina para predecir y responder a los problemas del mundo externo.

Las primarias características de la representación de conocimiento se expresan en dos formas:

- Cual información es necesariamente expuesta.
- Como la información es físicamente codificada para su uso.

El mayor reto para una red neuronal es aprender y adaptar un modelo del mundo real, y mantener la consistencia del mismo para lograr resolver las metas de su interés. El conocimiento del mundo consiste en dos tipos de información:

- El estado actual, o *información base o previa.*
- Las observaciones del entorno, obtenida por sensores diseñados para analizar el entorno. Estas pueden presentar ruido o errores, pero por lo general, estas observaciones son obtenidas para proveer un pozo de información, para extraer ejemplos utilizados para entrenar la red y así, reducir el error.

El entrenamiento de la red consiste en buscar conjuntos de pares de información de entrada y salida, compactado en el concepto de *información de entrenamiento* o *pruebas*. Para el diseño de la red neuronal, se debe seguir los siguientes pasos:

- *Aprendizaje*: Una arquitectura apropiada es seleccionada para la red, con la cantidad de nodos de entrada equivalente a los pixeles de la imagen, y una capa de salida de diez neuronas (una por cada número). Un conjunto de ejemplos es usado para entrenar la red por medio de un algoritmo compatible con dicha estructura de la red.
- *Testeo*: El reconocimiento de la red entrenada es puesta a prueba con información nueva no antes ingresada en la red. Es decir, la imagen de un dígito es ingresada en la red, pero no se le revela explícitamente la

identidad del dígito. Cuando la red acierta en identificar el digito correcto se puede afirmar que la red está *generalizando* la información aprendida.

En una red neuronal, la representación de la información de su entorno es definida por los valores extraídos por los parámetros libres de la red. La forma que toma dicha información constituye el diseño integro de la red y, por ende, de su desempeño. El concepto de *como* la información de la red es actualmente representado dentro de la misma es un tema complicado. Sin embargo, existen cuatro reglas generales acerca de la representación de la red:

1. *Similares ingresos de similares fuentes usualmente producen similares representaciones dentro de la red.*

2. *Objetos categorizados como diferentes clases deben tener diferentes representaciones dentro de la red.*

3. *Si una característica en particular resulta ser importante, entonces una mayor cantidad de neuronas deben estar involucradas en la representación de la misma.*

   Considerando, por ejemplo, en la detección de un radar en la presencia de interferencias como árboles, ruido, entre otros. El rendimiento de detección de dicho radar esta medida en dos términos de probabilidad:

   - *Detección*: Cuando la detección es verdadera.
   - *Falsa alarma*: Cuando se percibe una falsedad.

Según el criterio de Neyman-Pearson, la probabilidad de detección es maximizada cuando el valor de la restricción de la falsa no supera un valor previamente determinado por el experto. Para ello, debe haber una mayor cantidad de neuronas involucradas en la toma de decisión. La razón detrás de ello consiste en que la mayor cantidad de neuronas asegura un mayor grado de certeza en la toma de decisiones y tolerancia con respecto a neuronas propensas a fallas.

4. *Información previa e invariancias deben estar ingresadas dentro del diseño de la red siempre cuando sea posible para simplificar el diseño de la red.*

Esta regla es particularmente importante porque al cumplirla, el resultado consiste en una red neuronal con una estructura especializada. Esto es ventajoso debido a diversas razones:

- Redes visual y auditivamente biológicas son conocidas por ser altamente especializadas.
- Una red neuronal con estructura especializada usualmente tiene una menor cantidad de parámetros libres disponibles para reajustes comparado con una red completamente conectada. Consecuentemente, la red especializada requiere una menor cantidad de datos para su entrenamiento, aprende más rápido y regularmente generaliza mejor.
- El flujo de la transmisión de información a través de la red especializada es acelerado.
- El costo para construir una red especializada es reducido debido a su menor tamaño comparado con la red completamente conectada.

Cabe notar que al incorporar conocimientos previos a la red *restringe* la aplicación de esa red al particular problema que se busca resolver con dichos conocimientos.

- Como ingresar información previa al diseño de la red:

  En particular, se pueden utilizar dos técnicas:

  - Restringiendo la arquitectura de la red, usando conexiones locales conocidas como campos receptivos.
  - Limitando la selección de los pesos sinápticos.

  Ambas técnicas tienen el efecto secundario de reducir la cantidad de parámetros a ingresar en la red significantemente.

- Como ingresar invariancias al diseño de la red:

Existen tres técnicas para volver a la red invariante a las transformaciones:

- *Invariante por estructura*: Una red puede resultar ser invariante si su estructura es diseña apropiadamente. Es decir, la red puede estar diseñada para que las conexiones entre las neuronas sean creadas de tal manera que las versiones transformadas del mismo ingreso produzcan la misma señal de salida.

- *Invariante por entrenamiento*: Una red neuronal tiene la capacidad de reconocer patrones, la cual puede ser explotada para que la red reconozca un objeto en base a los ejemplos presentados en su entrenamiento. Sin embargo, esto conlleva el problema que la red reconocerá otros objetos de diferentes clases de la misma manera invariante.

- *Invariante por espacio permitido*: La tercera técnica consiste en extraer *características clave* que definen al contenido esencial de la información ingresada y que son invariantes a la transformación del ingreso. Esto ofrece tres ventajas: primero, que el número de valores a ingresar en la red se ve reducir a nivel realistas; los requerimientos impuestos en el diseño de la red se ven relajadas; y la invariancia para todos los objetos con respecto a las transformaciones conocidas es asegurada.

2.7.7 Perceptrón multicapa

2.7.7.1 Conceptos preliminares:

Los siguientes tres puntos resaltan las características del perceptrón multicapa:

- El modelo de cada neurona incluye una función de activación no linear *diferenciable* del resto.

- La red contiene una o más capas que están ocultas.

- La red exhibe un alto grado de conectividad, determinada por sus conexiones sinápticas.

Otro concepto importante es la forma de entrenamiento de los perceptores multicapa, el cual se logra mediante el algoritmo de *back-propagation*. Este procede en dos fases:

1. En la fase de forward, los pesos sinápticos son definidos previamente y la señal de salida es propagada a través de la red, capa por capa, hasta que llega hasta la capa de salida.

2. En la fase de backward, una señal de error es producida al comparar la salida de una red con su respuesta esperada. Esta señal es luego propagada por toda la red en manera inversa. En esta fase, los pesos de las neuronas son reajustadas para reducir el nivel de error. El cálculo de los ajustes resulta más complicado en las capas ocultas en comparación a la capa de salida.

Un concepto importante a detallar es acerca de la función de las neuronas ocultas, las cuales actúan como *detectores* de características. A medida que el proceso de aprendizaje se propaga a través del red multicapa, estas neuronas empiezan a "descubrir" las características claves que caracterizan a dicha información de entrenamiento por medio de la transformación no linear de la información en un *espacio de categorización de características*. Al tener una base de clasificar e identificar los patrones de la información, esta se vuelve más fácil de identificar para el perceptrón.

2.7.7.2 Algoritmo de propagación en backwards (Figura 28):

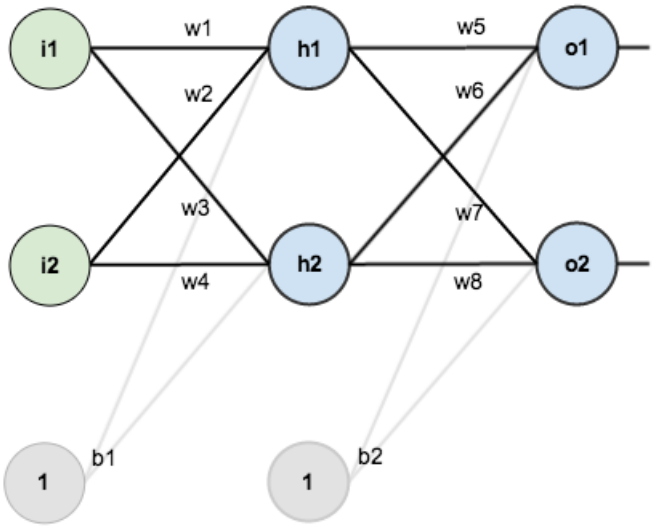


Figura 28: Red multicapa simple con 1 capa escondida.

La propagación en backwards es un proceso común para entrenar una red neuronal. Como se tiene conocimiento previo, la red neuronal puede tener entre una cantidad N de capas ocultas, una capa de entrada y una capa de salida, dependiendo de la red. Para el algoritmo de propagación en backwards, se tiene que tener en cuenta el resultado total de la última capa de la red, es decir, de la señal de error de la red. Para tener a mano algunos números a trabajar, aquí está la misma red, pero con los pesos, bias e inputs iniciales (Figura 29):

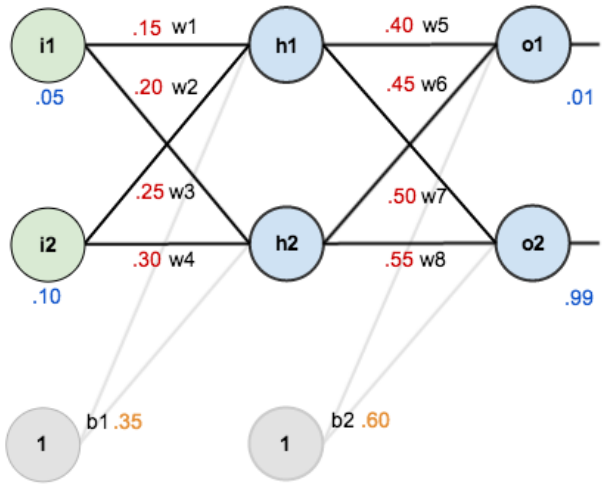


Figura 29: Red multicapa simple con 1 capa escondida con pesos e inputs.

El objetivo principal de la propagación en backwards es optimizar los pesos de las neuronas para que la red neuronal pueda corregir arbitrariamente los inputs para obtener el resultado esperado.

Se puede definir como el cálculo de la señal de error de una iteración *n* como:

$$e_j(n) = d_j(n) - y_j(n) \text{ siendo } j \text{ el nodo de salida. } Ecuación\ (11)$$

El valor instantáneo del resultado de error de cada neurona se puede representar como:

$$\frac{1}{2}e_j^2(n) \qquad Ecuación\ (12)$$

Y como la red es un conjunto de neuronas interconectadas, para calcular el error total se usa la siguiente fórmula:

$$\mathcal{E}(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \qquad Ecuación\ (13)$$

Donde C es el número de neuronas de la capa de salida. Después de ello, dependiendo de la cantidad de patrones ejemplo (*N*) en el set de entrenamiento, el cual se representa así:

$$\varepsilon_{av} = \frac{1}{N}\sum_{n=1}^{N} \varepsilon(n) \qquad Ecuación\ (14)$$

Teniendo en claro el resultado del error total de la red, el objetivo del proceso de aprendizaje es de reducir dicho valor lo más posible por medio de diversas iteraciones. Para lograr este proceso, se debe reajustar los valores de los pesos de las neuronas en relación a los resultados a través de la red.

La parcial derivativa $\partial\mathcal{E}(n)/\partial w_{ji}(n)$ representa el *factor de sensibilidad* del peso w_{ji}, y la gradiente resultante se puede expresar como:

$$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathcal{E}(n)}{\partial e_j(n)} x \frac{\partial e_j(n)}{\partial y_j(n)} x \frac{\partial y_j(n)}{\partial v_j(n)} x \frac{\partial v_j(n)}{\partial w_{ji}(n)} \qquad Ecuación\ (15)$$

Usando las fórmulas anteriores, se puede descomponer la previa fórmula a:

$$\frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'\left(v_j(n)\right) y_i(n) \qquad Ecuación\ (16)$$

La corrección aplicada al peso $w_{ji}(n)$ es definida por la regla *delta*:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial w_{ji}(n)} \qquad Ecuación\ (17)$$

η en este contexto es el *parámetro del ratio de aprendizaje* del algoritmo de baja propagación. El uso del signo menos denota una *tendencia descendente del peso*.

Generalmente es entre 0 y 1. Si se tiene un ratio muy alto, los cambios en w serán más bruscos pero se corre el riesgo de que el algoritmo pase por alto los valores óptimos de *w*. Si se tiene un ratio muy bajo, habrá más iteraciones para actualizar *w* pero puede que el algoritmo se demore mucho más en encontrar los valores óptimos de *w*. En combinación con la fórmula de la gradiente, la regla delta se vuelve en:

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \qquad Ecuación\ (18)$$

Con la gradiente δ_j(n) definida como:

$$\begin{aligned} \delta_j(n) &= -\frac{\partial \mathcal{E}(n)}{\partial v_j(n)} \qquad Ecuación\ (19) \\ &= -\frac{\partial \mathcal{E}(n)}{\partial e_j(n)} x \frac{\partial e_j(n)}{\partial y_j(n)} x \frac{\partial y_j(n)}{\partial v_j(n)} \\ &= e_j(n)\varphi_j'\left(v_j(n)\right) \end{aligned}$$

Existen, sin embargo, la duda de en qué grado penalizar o premiar a una neurona, dependiendo si es una neurona oculta o de salida. Para ello, se define dos casos de modificación de los pesos:

- Cuando la neurona *j* está ubicada en la capa de salida de la red, se puede usar la señal de error $e_j(n)$. Al tener dicho dato, se es posible computar la gradiente local $\delta_j(n)$ para dicho nodo.

- Cuando la neurona j es un nodo de la capa oculta, no existe una respuesta específica para la neurona. La señal de error para una neurona escondida tiene que ser determinada recursivamente y trabajando en calcular todas las señales de error de todas las neuronas conectadas a esa neurona en específico. Para su cálculo se debe redefinir la gradiente $\delta_j(n)$ (para la neurona j) como:

$$\delta_j(n) = -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \; x \; \frac{\partial y_j(n)}{\partial v_j(n)} \qquad Ecuación\ (20)$$
$$= -\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} \varphi_j'\left(v_j(n)\right)$$

Para calcular la derivativa parcial $\frac{\partial \mathcal{E}(n)}{\partial y_j(n)}$, se debe recordar que:

$\mathcal{E}(n) = \frac{1}{2}\sum_{k\in C} e_k^2(n)$ siendo k la neurona de salida

Diferenciando esta ecuación en relación a la señal de función $y_j(n)$ se obtiene (Figura 30):

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)} \qquad Ecuación\ (21)$$

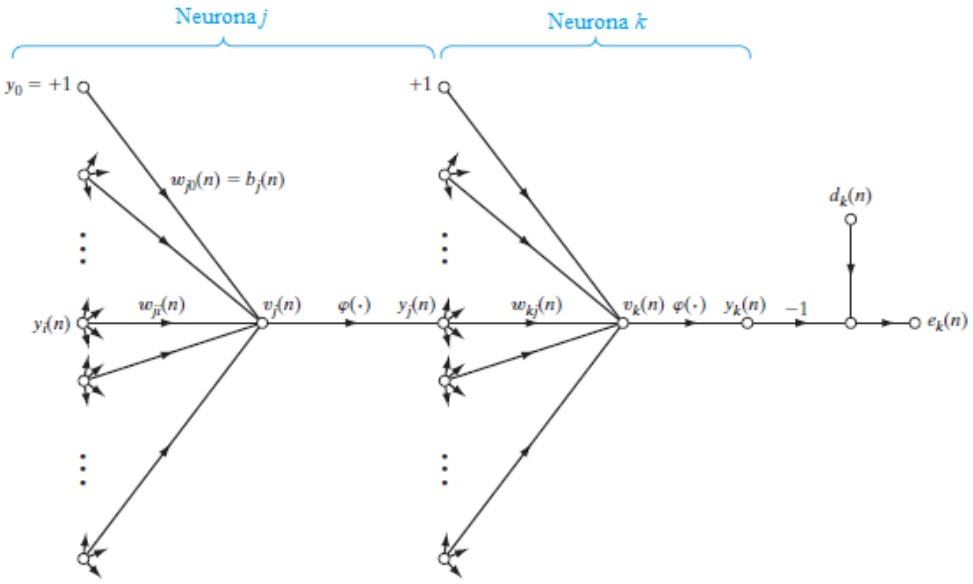


Figura 30: Grafico de una señal de un solo flujo detallando la salida de la neurona k conectada a la neurona oculta j.

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

Luego, se utiliza la regla en cadena de para la parcial derivativa y se reescribe la formula a su equivalente:

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} \; x \; \frac{\partial v_k(n)}{\partial y_j(n)} \qquad Ecuación\ (22)$$

Sin embargo, se da a notar que:

$$e_k(n) = d_k(n) - \; y_k(n)$$

$$= \; d_k(n) - \; \varphi_k\big(v_k(n)\big) \qquad Ecuación\ (23)$$

$$\frac{\partial e_k(n)}{\partial v_k(n)} = \; -\varphi_k'\big(v_k(n)\big) \qquad Ecuación\ (24)$$

Por ende:

$$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n) \qquad Ecuación\ (25)$$

…donde *m* es el número total de ingresos (sin incluir al bias) aplicada a la neurona *k*. Diferenciando las ecuaciones con respecto a $y_j(n)$ tenemos:

$$\frac{\partial v_k(n)}{\partial y_j(n)} = \; w_{kj}(n) \qquad Ecuación\ (26)$$

Teniendo en cuenta las formulas obtenidas, se produce a obtener la deseada derivativa parcial:

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \; -\sum_k e_k(n)\, \varphi_k'\big(v_k(n)\big) w_{kj}(n)$$

$$= \; -\sum_k \delta_k(n)\, w_{kj}(n) \qquad Ecuación\ (27)$$

Finalmente, se puede deducir la fórmula de propagación en backwards para la gradiente local $\delta_j(n)$, descrita como:

$$\delta_j(n) = \varphi'_j\left(v_j(n)\right)\sum_k \delta_k(n)\, w_{kj}(n), \qquad Ecuación\ (28)$$

con *j* como la neurona escondida

En resumen, la relación de todos los elementos de la propagación en backwards se puede entender con el siguiente gráfico:

$$\Delta\, w_{ji}(n) = \eta\, x\, \delta_j(n)\, x\, y_i(n) \qquad Ecuación\ (29)$$

…donde $\Delta\, w_{ji}(n)$ es la corrección de los pesos; η el parámetro de aprendizaje; $\delta_j(n)$ como la gradiente local y $y_i(n)$ como la señal de entrada.

En la aplicación del algoritmo de propagación, existen dos pases distintos de operación de procesos:

- Pase en *forward*:
    - El patrón de actividades (los *inputs*) son aplicados a los nodos sensoriales de red, propagándose capa por capa, hasta que finalmente un set de *outputs* es producido como respuesta de la red. Durante este proceso, los pesos de las sinapsis entre nodos son fijas.
    - En este modo, los pesos sinápticos permanecen inalterados a través de la red, y el patrón de señales de la red son operadas de neurona a neurona, de izquierda a derecha. Es decir, el proceso lógico de la red empieza desde la primera capa oculta y termina en la capa de resultados por medio de la operación de cada neurona de esta capa.
- Pase en *backward:*
    - En esta propagación lo que se muestra es una señal de error en base a una deseada respuesta de la red. Como lo dice su nombre, la propagación en *backward* parte en sentido inverso a través de la red, y los pesos sinápticos son ajustados para volver cada vez el resultado actual de la red más cerca al resultado deseado en un sentido estadístico.
    - El proceso en backward es lo opuesto. Empezando desde la red de resultado hacia el origen, pasando las señales de error capa por capa de cada neurona

en la red. Es este proceso el cual permite a los pesos sinápticos cambiar para acerca cada vez a los pesos óptimos.

El algoritmo de propagación en backwards provee una aproximación de la red correcta. Por medio de la modificación de los patrones del ratio de aprendizaje, se busca llegar a crear una red óptima que muestre un ratio aceptable de aprendizaje, sin sacrificar estabilidad alguna de la misma.

2.7.7.3 Ratio de aprendizaje:

El algoritmo de propagación en backwards provee una aproximación de la trayectoria de la regulación de los pesos de la red. Mientras más pequeño es el ratio de aprendizaje, menores serán los cambios en los pesos de la red a través de las neuronas interconectadas, volviendo la trayectoria de variación del peso más fluida. Pero esto ocurre con el costo de volver el aprendizaje muy lento, y si por el otro lado el parámetro de aprendizaje es muy largo, los cambios de los pesos pueden resultar ser muy bruscos que es posible que la red colapse.

Una manera de regular dicho ratio de manera de evitar la inestabilidad de la red consiste en modificar la regla delta a:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_j(n) \qquad Ecuación\ (30)$$

…donde α es usualmente un número positivo llamado la *constante del momentum*. Para vez la secuencia de presentaciones de patrones de los pesos sinápticos, se debe reformular la ecuación anterior como una secuencia de tiempo con el índice *t*, que fluye desde 0 a n. Este controla el comportamiento del bucle alrededor de $\Delta w_{ji}(n)$ como se muestra en la Figura 31.

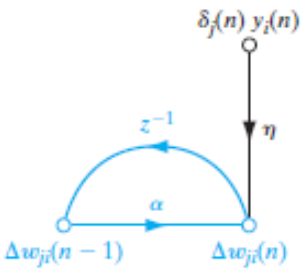


Figura 31: Flujo de señal de una sola vía ilustrando el efecto de α

Fuente: Haykin, S., 2009. *Neural networks and Learning Machines*

En base a la ecuación anterior, se puede definir la ecuación de corrección de los pesos ($\Delta w_{ji}(n)$) como:

$$\Delta w_{ji}(n) = \eta \sum_{t=0}^{n} \alpha^{n-t} \delta_j(t) y_i(t) \qquad Ecuación\ (31)$$

…el cual representa la secuencia de tiempo con una duración de n+1. Se puede también deducir que el producto $\delta_j(n) y_j(n)$ es igual a $-\frac{\delta \varepsilon(n)}{\delta w_{ji}(n)}$. Con esto en consideración, la ecuación puede ser rescrita a:

$$\Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} \alpha^{n-t} \frac{\delta \varepsilon(t)}{\delta w_{ji}(t)} \qquad Ecuación\ (32)$$

Basado en esta relación, podemos hacer las siguientes relaciones:

- El ajuste de $\Delta w_{ji}(n)$ representa la suma de las secuencias de tiempo exponencialmente pesadas.
- Cuando la parcial derivativa $\frac{\delta \varepsilon(t)}{\delta w_{ji}(t)}$ tiene el mismo signo algebraico en iteraciones consecutivas, la suma exponencial $\Delta w_{ji}(n)$ crece en magnitud, y consecuentemente, también el peso $w_{ji}(n)$. Es decir, el efecto de incluir momentum en la propagación en backwards es *acelerante* sobre los pesos.
- En cambio, cuando la parcial derivativa posee diferente signos algebraicos a través de iteraciones, $\Delta w_{ji}(n)$ decrece al igual que sus pesos. Es decir, el efecto de incluir momentum en la propagación en backwards posee un efecto *estabilizador* que oscila entre signos.

2.7.7.4 Criterio para detener el algoritmo:

En general, el algoritmo de propagación en backwards no puede revelar en que momento empieza a converger y no posee criterios bien definidos para detener su operación. Existen, sin embargo, medidas para terminar los reajustes de los pesos. Para formular dichos criterios, es necesario definir un rango mínimo global o local para el error. Una vez tenido esto en

consideración, se puede empezar a formular un criterio de convergencia para el aprendizaje por propagación en backwards (Kramer y Sangiovanni-Vincentelli,1989);

"El algoritmo de propagación en backwards es considerado que ha convergido cuando la norma euclidiana del vector gradiente alcanza un límite de gradiente suficientemente pequeño."

La desventaja de este criterio de convergencia consiste en que, en pruebas exitosas, el aprendizaje puede tomar mucho tiempo. Por ello, se puede sugerir tomar en cuenta otro criterio de convergencia para este caso:

"El algoritmo de propagación en backwards es considerado que ha convergido cuando el ratio absoluto del cambio del error estándar por época es lo suficientemente pequeño."

El ratio es considerado pequeño si existe en un rango de 0.1 a 1 por ciento por época de aprendizaje. Desafortunadamente, el uso de este criterio pequeño puede resultar en una terminación prematura del proceso de aprendizaje.

Alternativamente, se le puede decir a la red que realice una prueba de generalización por cada iteración de aprendizaje. Con ello, se puede controlar el proceso de aprendizaje para que se detenga cuando el desempeño de la generalización alcanza su máximo punto.

2.7.7.5 Teoremas importantes:

Un teorema que se toma en consideración del aprendizaje de la red fue el *teorema de Kolmogorov* , el cual sostiene que cualquier función continua $f: [0,1]^n \Rightarrow R^m, f(x) = y$, f puede ser implementada por una red neural de tres capas sin retroalimentación que tiene una capa de entrada de *n* elementos que únicamente copian las entradas a la siguiente capa, (2n+1) en la capa intermedia y *m* neuronas en la capa de salida. (Valenzuela, M., 1995)

Los elementos de la capa intermedia poseen la siguiente función:

$$z_k = \sum_{j=1}^{n} \lambda^k \psi(x_j + k\epsilon) + k, \qquad Ecuación\ (33)$$

…donde la constante real λ y la función continua real monótona creciente ψ son independientes de *f*. La constante ϵ es un numero racional entre 0 y δ, donde δ es una constante positiva escogida arbitrariamente.

Los elementos de la capa de salida tienen la siguiente representación matemática:

$$y_i = \sum_{k=1}^{2n+1} g_i(z_k), \qquad Ecuación\ (34)$$

Siendo las funciones g_i continuas y reales, que dependen de *f* y ϵ.

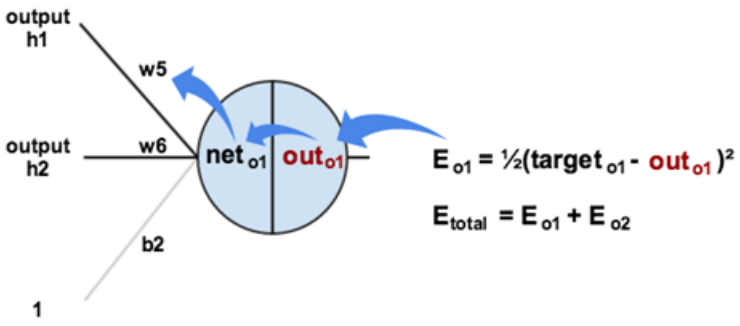
2.7.8 Ejemplo simple:

Regresando al ejemplo del gráfico de la Figura 28, se toma en consideración *w*5. Se quiere saber cuánto afecta un cambio en el peso *w*5 en el error total, es decir, $\frac{\partial Etotal}{\partial w5}$.

Aplicando la regla de cadena, se obtiene:

$$\frac{\partial E_{total}}{\partial w5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w5} \qquad Ecuación\ (35)$$

Visualmente, lo que está sucediendo es:



Con la información presentada, se pregunta cuanto cambia el error total en relación a la salida total:

$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2 \qquad Ecucación\ (36)$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = 2 * \frac{1}{2}(target_{o1} - out_{o1})^{2-1} * -1 + 0$$

$$\frac{\partial E_{total}}{\partial out_{o1}} = -(target_{o1} - out_{o1}) = -(0.01 - 0.75136507) = 0.74136507$$

Luego, se pregunta cuanto varia la entrada $o1$ en relación al input total neto:

$$out_{o1} = \frac{1}{1 + e^{-net_{o1}}} \qquad Ecucación\ (37)$$

$\frac{\partial out_{o1}}{\partial net_{o1}} = out_{o1}(1 - out_{o1})$ = 0.75136507 (1-075136507) = 0.186815602

Finalmente, se calcula cuanto del input total de $o1$ cambia con respecto de $w5$:

$$net_{o1} = w5 * out_{h1} + w6 * out_{h2} + b_2 * 1 \qquad Ecucación\ (38)$$

$$\frac{\partial net_{o1}}{\partial w5} = 1 * out_{h1} * w5^{1-1} + 0 + 0 = out_{h1} = 0.593269992$$

Combinando todas las fórmulas:

$$\frac{\partial E_{total}}{\partial w5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w5}$$

$$\frac{\partial E_{total}}{\partial w5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

Para reducir el error, se substrae el valor del actual peso (opcionalmente multiplicado por el ratio de aprendizaje, que se colocará a 0.5):

$$w5^{+} = w5 - \eta * \frac{\partial E_{total}}{\partial w5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

Esta fórmula se puede repetir para hallar los nuevos pesos de $w6$, $w7$ y $w8$:

$$w6^{+} = 0.40866186$$

$$w7^{+} = 0.511301270$$

$$w8^{+} = 0.561370121$$

2.8 Beneficios:

Haykin, S. (2009) afirma que el uso de las redes neuronales ofrece las siguientes propiedades y capabilidades:

1. No linearidad: Una red neuronal es por naturaleza no linear, lo cual resulta importante para procesar información de naturaleza también no lineal (patrones de habla).

2. Mapeo de *input-output*: Consiste en la facilidad de modificar los pesos de la red neuronal para obtener la respuesta deseada del modelo, el cual forma parte del paradigma de aprendizaje llamado *aprendizaje supervisado.*

3. Adaptabilidad: Las redes neuronales tienen la capacidad de adaptar sus pesos sinápticos en relación a los cambios en su ambiente.

4. Respuesta evidencial: Una red neuronal puede ser diseñada para proveer información no solo acerca del patrón seleccionado, sino además de la confianza de la decisión tomada.

5. Información contextual: Cada neurona de la red es potencialmente afectada por la actividad global de las demás neuronas.

6. Tolerancia al fallo: La red es inherentemente resistente a fallas de rendimiento y puede operar correctamente bajo condiciones adversas.

2.9 Matriz de confusión:

Una matriz de confusión es una tabla que normalmente se usa para describir el rendimiento de un modelo de clasificación en relación de la veracidad de sus resultados.

Para demostrar la utilidad y funcionamiento de la matriz de confusión se va a presentar un ejemplo que busca identificar cuantos de los pacientes presentan una enfermedad X de una muestra de 165 personas:

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Esta matriz muestra la siguiente información:

- Existen 2 estados: si y no, el cual determina si el paciente posee o no la enfermedad.
- De todos los casos, se predijo “si” 110 veces y “no” 55 veces.
- En realidad, 105 pacientes poseen la enfermedad y 60 no.

Con esta información, se puede determinar:

- Verdaderos positivos: Cuando se predijo si y poseen la enfermedad.
- Verdaderos negativos: Cuando se predijo no y no poseen la enfermedad.
- Falsos positivos: Cuando se predijo sí, pero no poseen la enfermedad.
- Falsos negativos: Cuando se predijo no, pero si poseen la enfermedad.

Con estos datos, se busca comprobar ciertas características del modelo:

- **Exactitud:** ¿Qué porcentaje de veces la red es exacta?
- **Ratio de error:** ¿Cuantas veces se ha equivocado?
- **Precisión:** ¿Cuándo se predice una veracidad, que tan exacto o cierto es? ¿Es decir, del total de predicciones en verdadero, cuanto porcentaje fue verídico?

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

- **Exactitud:**

(TP+TN)/total = (100+50)/165 = **0.91** *Ecuación* (39)

- **Ratio de error:**

(FP+FN)/total = (10+5)/165 = **0.09** *Ecuación* (40)

- **Precisión:**

TP/predicted yes = 100/110 = **0.91** *Ecuación* (41)

2.10 Hipótesis:

Este proyecto tiene el objetivo de demostrar la efectividad del uso de una plataforma de videojuegos para la educación de usuarios de un rango de edad de 5 a 11 años acerca del vocabulario de idioma inglés. Para sustentar la razón de este proyecto, se busca validar la relación positiva entre jugar el videojuego y el nivel educativo de los usuarios/alumnos. Como se va a efectuar la medida de los errores y éxitos de los usuarios al experimentar con el aplicativo, se espera conseguir datos verídicos para comprobar las hipótesis mostradas a continuación.

H1: Existe una relación directa entre la inclusión de dificultad de los niveles con el aprendizaje del usuario.

H2: Existe una relación directa entre la inclusión de pistas audiovisuales en el videojuego con el aprendizaje del usuario.

H3: Existe una relación directa entre la cantidad de palabras acertadas por el usuario y el impacto de la enseñanza del juego.

A partir de estas hipótesis se espera aportar en la evaluación de la implementación del ámbito educativo en los videojuegos para el aprendizaje del vocabulario en inglés. Además, se pretende ampliar el entendimiento del potencial de los videojuegos y su relación con la efectividad del aprendizaje y del impacto del constante estudio por medio del aplicativo.

2.11 Algebra lineal

2.11.1 Multiplicación de matrices:

Sea **A** una matriz de m×n y **B** una matriz de n×p. Entonces el producto **AB** es la matriz m×p **C**, cuyo elemento cij de la i-ésima fila y la j-ésima columna se obtiene de la siguiente manera: sumar los productos formados al multiplicar en orden, los elementos de la iésima fila de **A** por el elemento correspondiente (es decir el primero, el segundo, etc), de la j-ésima columna de **B**.

Para efectuar la multiplicación, es necesario que el número de columnas de A sea igual al número de filas de B. La matriz resultante, por ello, tendrá la misma cantidad de filas que la matriz A y la misma cantidad de columnas de la matriz B. Por ejemplo, si se tiene una matriz de 2x3 y se multiplica por una de 3x2, la matriz resultante será una de 2x2.

$$\begin{bmatrix} 3 & 1 & 4 \\ 1 & 3 & 7 \end{bmatrix} * \begin{bmatrix} 3 & 2 \\ 1 & 13 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

a = 3*3 + 1*1 + 4*2 = 18

b = 3*2 + 1*13 + 4*1 = 23

c = 1*3 + 3*1 + 7*2 =21

d = 1*2 + 3*13 + 7*1 = 48

$$= \begin{bmatrix} 18 & 23 \\ 21 & 48 \end{bmatrix}$$

Capitulo III: METODOLOGÍA:

3.1 Diseño de la Investigación

3.1.1 Diseño

La presente investigación es experimental al tener control en las variables y poder analizar su relación. Es una investigación longitudinal al buscar evaluar la mejora del nivel de aprendizaje del público objetivo a través del paso de un periodo considerable de tiempo.

3.1.2 Tipo

La investigación es de tipo de desarrollo tecnológico debido a la elaboración de un producto conciso que busca evaluar y mejorar el nivel del conocimiento del vocabulario del idioma inglés.

3.1.3 Enfoque

La investigación tendrá un enfoque cuantitativo al estar centrada en variables numéricas que apoyen en establecer las relaciones planteadas.

3.1.4 Descripción del Prototipo de Investigación

El prototipo de videojuego consiste en dos elementos principales. El primero es el videojuego en sí, en formato apk para la instalación en dispositivos compatibles con Android. El segundo es una base de datos simple la cual contiene los elementos a cargar en el videojuego para crear un escenario agradable para reforzar el aprendizaje del idioma ingles en el usuario.

La decisión de incluir el archivo de videojuego de la base de datos con la base de datos consistió en la consideración de la velocidad de acceso de la información. Además, al tener la base de datos separada del archivo se vuelve más simple actualizar dicha base sin que exista una gran necesidad de sacar varias versiones del apk si es que se busca actualizar la base de datos (Figura 32).

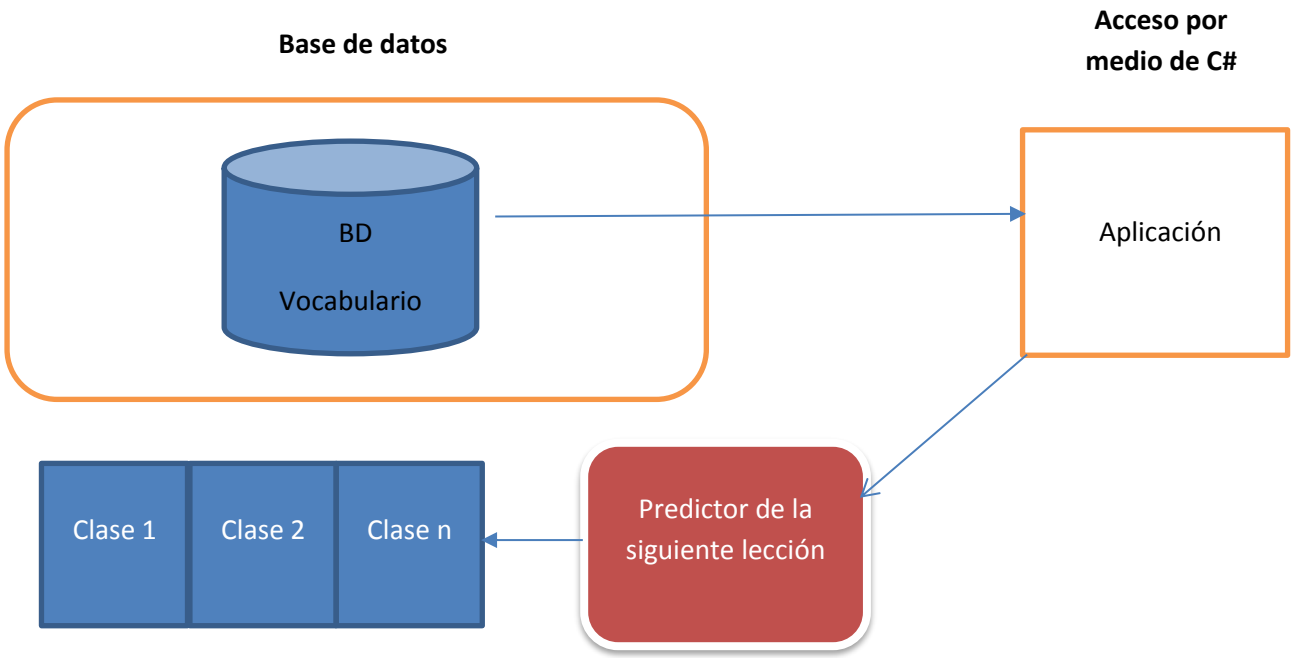


Figura 32: Descripción grafica de la manipulación de data de la base de datos (creación propia)

En relación de los elementos necesarios para el funcionamiento del producto final, se tiene en consideración estos detalles.

- Por el momento y por motivos de prueba, la aplicación de videojuego es compatible solo con Android.
- El almacenamiento o base de datos está compuesto por la base de datos SQLite y manejado por el aplicativo del SQLite Manager de Firefox.
- La extracción, registro y modificación de valores de la base de datos es manejada internamente dentro del juego, por medio de scripts C#.

3.1.4.1 Secuencia de *gameplay* del usuario:

La experiencia del videojuego que el usuario va a experimentar se define en la siguiente secuencia que se explicara a continuación (Figura 33):

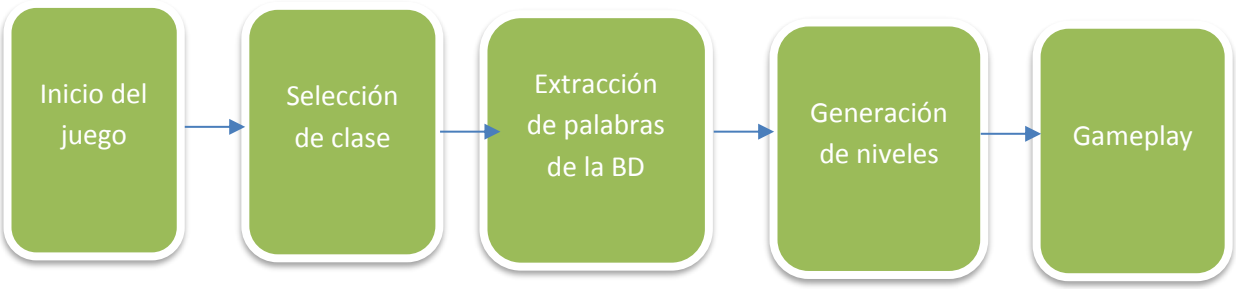


Figura 33: Secuencia de gameplay (creación propia)

Inicio del juego:

Al iniciar el aplicativo, el usuario ve una secuencia de menú simple, el cual presenta las opciones de iniciar la aplicación, mostrar información del juego en sí y la opción de cerrar el aplicativo.

Selección de clase:

Una vez pasada dicha pantalla, se presenta una nueva escena en la cual se le da la opción al jugador de escoger la clase correspondiente que desea aprender. Cabe añadir que cada clase contiene una colección diferente de palabras en inglés las cuales varían en dificultad de aprendizaje en relación a la clase seleccionada al iniciar.

Extracción de palabras de la BD:

Una vez elegida la clase a escoger, la aplicación busca acceder a la base de datos para recoger las palabras que se van a utilizar en el aprendizaje en relación a la clase seleccionada por medio a una llamada a la base de datos usando scripts que contienen las sentencias de SQL. El usuario no ve dicha secuencia ya que se realiza implícitamente apenas la selección es hecha.

Generación de niveles:

Se generan los niveles correspondientes para que los usuarios busquen aprender nuevas palabras por medio del *gameplay*, el cual, como se ha sustentado en las partes anteriores, es una manera efectiva para la retención de nueva información por medio de crear retos y objetivos para que el usuario supere.

Gameplay:

El juego en si consiste en la recolección de letras que forman parte de una palabra que aparece brevemente en la pantalla, a la vez que se genera el sonido correspondiente de su dicción correcta. Después de haber pasado un lapso de tiempo correspondiente a la longitud de la palabra, la pantalla cambia, revelando el nivel seleccionado desde la pantalla de generación de niveles, con las letras apareciendo con otro arte y esparciéndose aleatoriamente a través de la pantalla.

El objetivo del jugador es de recolectar las letras en el orden correcto en relación a la palabra mostrada al inicio del nivel. En el caso de que el usuario piense que ocurrió un error, este puede corregir su error por medio de un botón al costado de la pantalla, a la vez que se le presenta la opción de escuchar la pantalla otra vez.

Una vez que el jugador sienta que su elección es correcta, este puede mandar su respuesta por medio de un botón y, dependiendo de cómo ha recolectado las letras, el juego le notifica al jugador si ha acertado o no con su elección. Después de completar diez veces la misma escena, se desbloquea el siguiente nivel, el cual posee diferente *layout* para que el jugador recolecte las palabras (Figura 34).

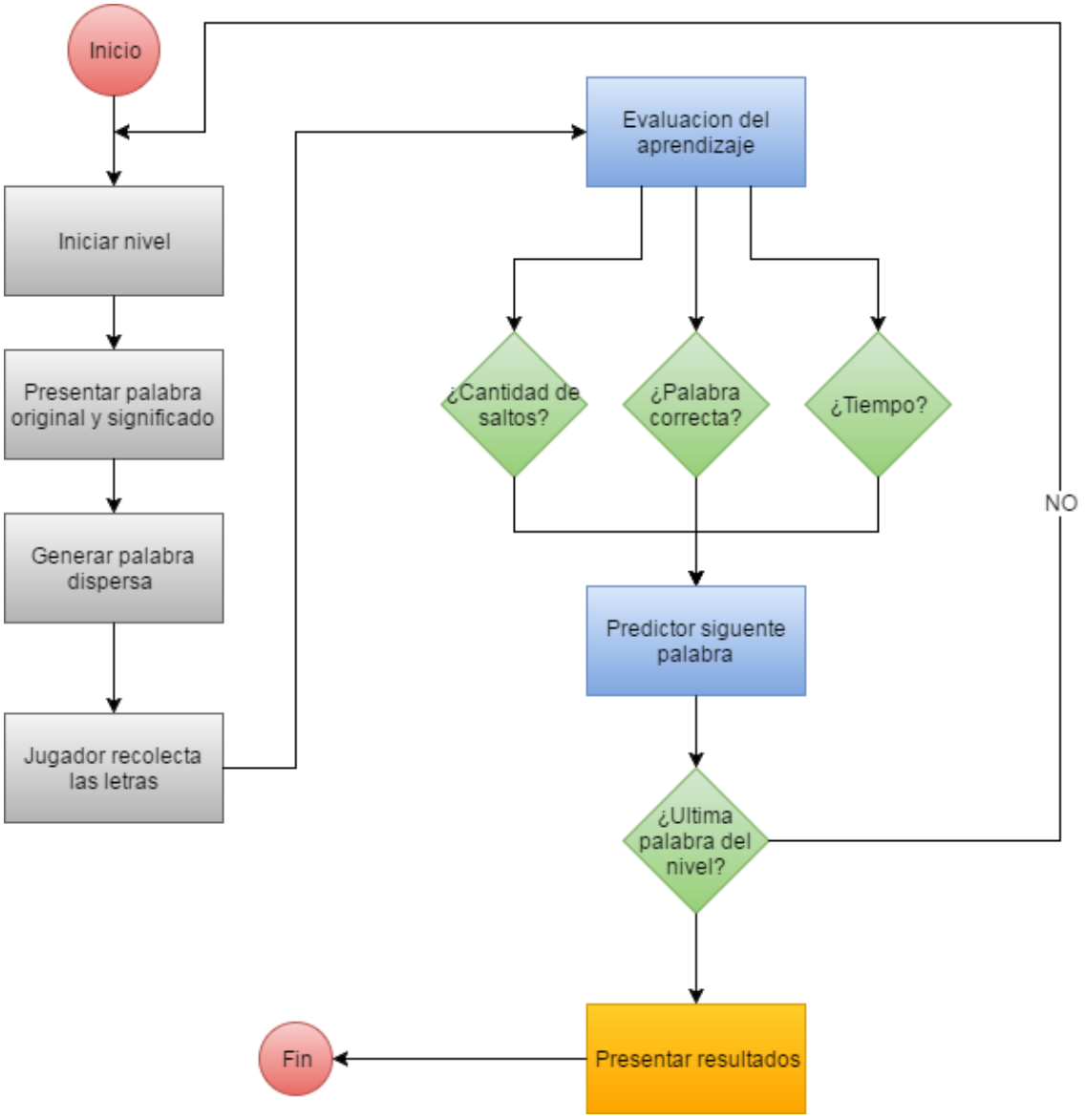


Figura 34: Secuencia de gameplay (creación propia)

3.1.4.2 Obtención de data de la base de datos:

Por medio de scripts internos codificados en C#, la aplicación tiene la capacidad de acceder y extraer información acerca de las clases de vocabulario, las palabras de las clases y de información relacionada a su traducción, imágenes relacionadas al significado e incluso sonidos acerca de la dicción de las palabras.

Después de extraer dichos datos, la aplicación genera sus propias listas para almacenar la data, procesa la información para generar los elementos del juego y,

finalmente, manifiesta dichos elementos en el juego en las instancias de *gameplay* para apoyar al inicio de proceso de aprendizaje.

3.1.4.3 Aplicación de la teoría de aprendizaje en el juego:

La aplicación del videojuego busca generar un aprendizaje a largo plazo de diversas palabras en inglés por medio de un método lúdico para agudizar el aprendizaje. De la selección de palabras escogidas por clases, estas se dividen en una cantidad de niveles determinada y, dependiendo del resultado del nivel, se repetirá la palabra en la siguiente instancia del nivel si el usuario ha fallado en dicha palabra hasta lograr un aprendizaje concreto de la palabra, basado en la forma de aprendizaje que aplicaba la aplicación del *Memrise,* como se defino en anterioridad.

Dicho algoritmo de selección de palabra se desarrolló y entreno en Octave, el cual es una interfaz web de usuario que permite utilizar Matlab, un ambiente numérico computacional multi-paradigma, para asegurar un nivel de aleatoriedad que se adapta al objetivo de aprendizaje del vocabulario de inglés que se busca con este proyecto.

3.1.5 Fases de la Investigación

La presente investigación tiene dos fases principales. La primera involucra la creación del prototipo funcional. La segunda involucra la recopilación de resultados y su posterior análisis. Se describirán ambas fases a detalle.

3.1.5.1 Desarrollo del prototipo

El desarrollo del prototipo de hizo en base a la metodología de desarrollo de videojuegos, tomando en consideración el objetivo de educación de vocabulario del idioma ingles del proyecto. Para esto se realizaron iteraciones de las etapas de análisis, diseño, construcción, pruebas y mantenimiento.

Análisis:

Para empezar, se investigo acerca de otros modelos similares de videojuegos con objetivos de aprendizaje. Como referencias principales se tomaron en cuenta aplicaciones como *Memrise* y Anki, el ultimo que implementa un algoritmo de repetición que prioriza

la urgencia de algunos objetos sobre otros (Anki (software),s.f). Ambos se enfocan en el lado de instrucción de vocabulario para sus usuarios, pero carecen el aspecto lúdico que este proyecto busca brindar para facilitar aún más el aprendizaje. Para el sustento de esta tesis, se indago acerca de aplicaciones de ingeniería de software en juegos, para encontrar una oportunidad de ámbito no explorado, y, en base a lo encontrado, se decidió implementar un algoritmo de predicción generado por una red neuronal, entrenada en Octave. Los datos de entrenamiento de la red fueron generados mediante la creación de una base de datos extrayendo una colección de 50 palabras de una página de las 2000 palabras más usadas del vocabulario inglés. (Top 2000 vocabulary words, 2005- 2017)

Diseño:

Se definió en empezar con el diseño del juego primero para realizar experimentos en relación de la aceptación del producto del mismo. Pararelamente, se empezó el diseño de la base de datos y la asignación de data al mismo dependiendo de las necesidades que el juego requería cumplir. Con eso en mente, se decidió diseñar el juego del género de plataformas, por su facilidad de atraer la atención del usuario y de su facilidad al programar. Para empezar, se empezó con diseñar la apariencia del personaje controlable, con el objetivo que sea apelable para el público objetivo. (Figura 35)

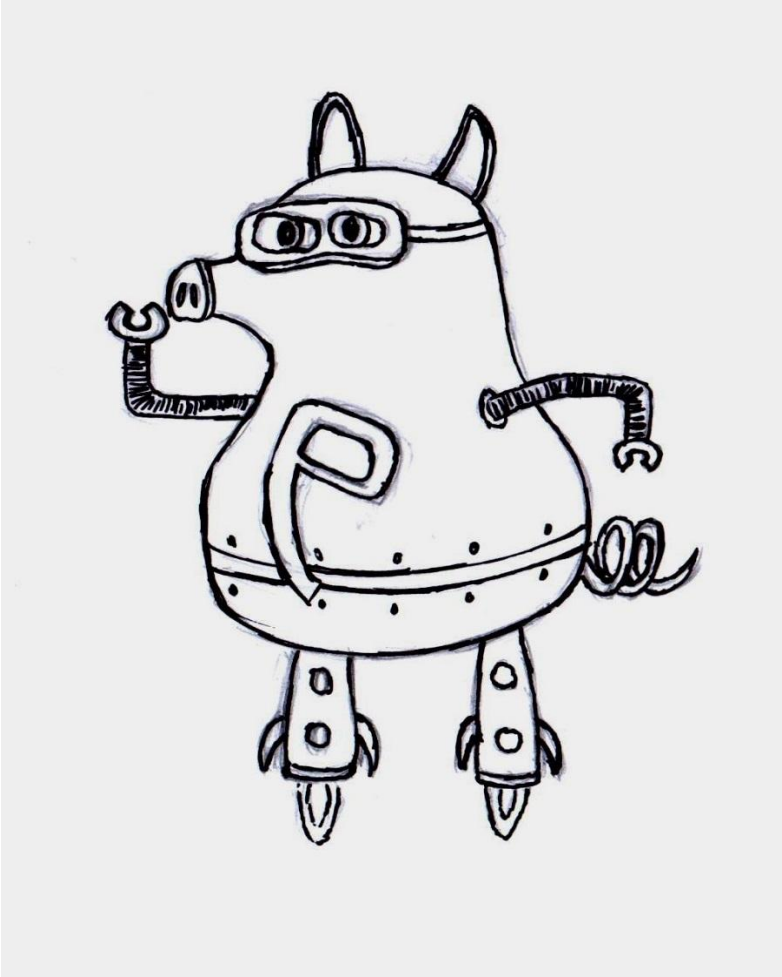


Figura 35: Bosquejo del personaje (creación propia)

Durante el planteamiento del diseño, se definió lanzar el producto para plataformas de Android debido a la popularidad de la plataforma en el mercado. Con esta idea enfocada, se realizaron bosquejos de cada pantalla y estudios en base al UX (User experience design) para encontrar el diseño correcto para el usuario (Figura 36):

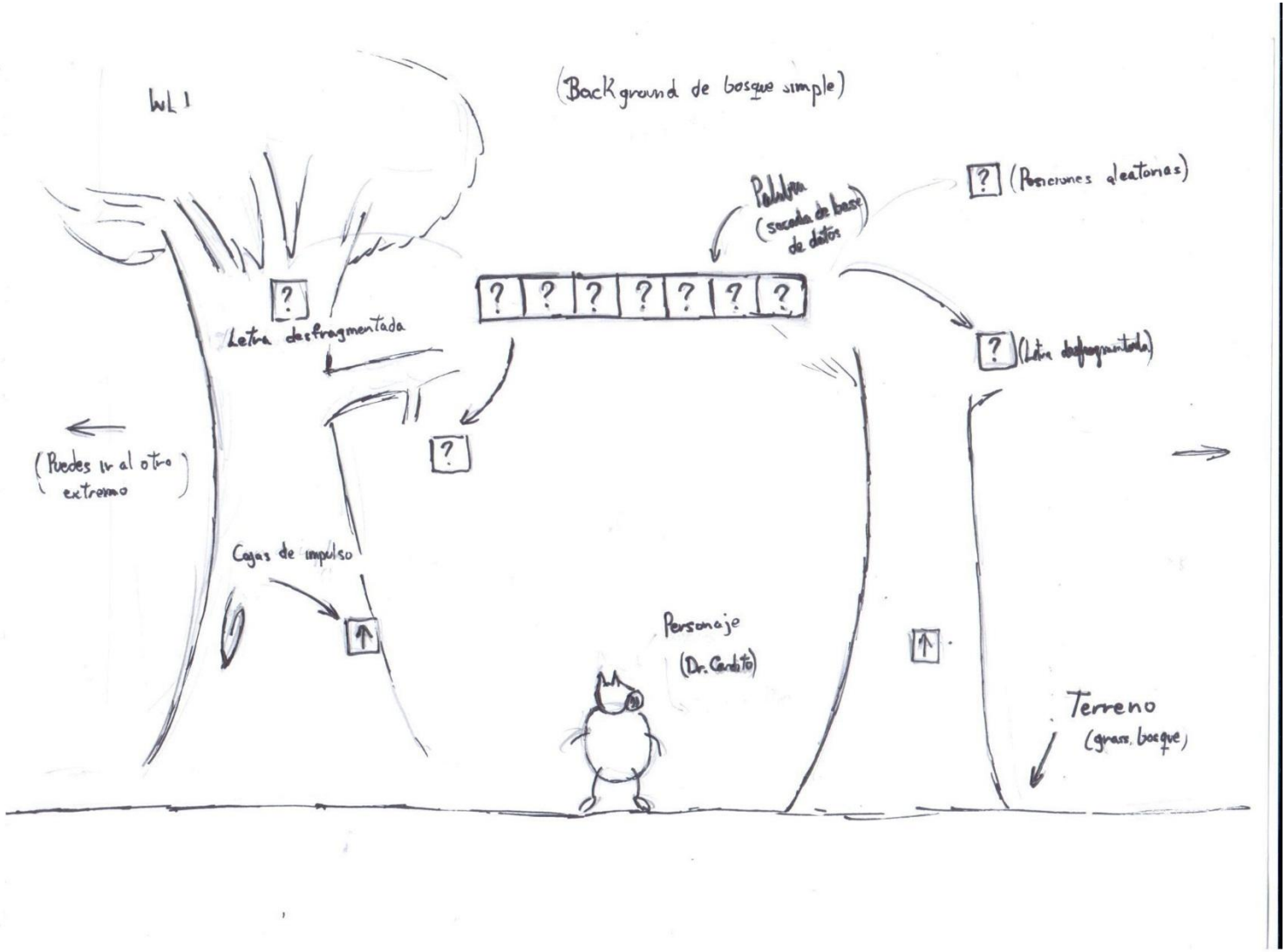


Figura 36: Bosquejos iniciales de UX (creación propia)

Relacionado a esto, se empezó el diseño de los posibles niveles a presentarse dentro del juego (Figura 37). Más detalles en anexos:

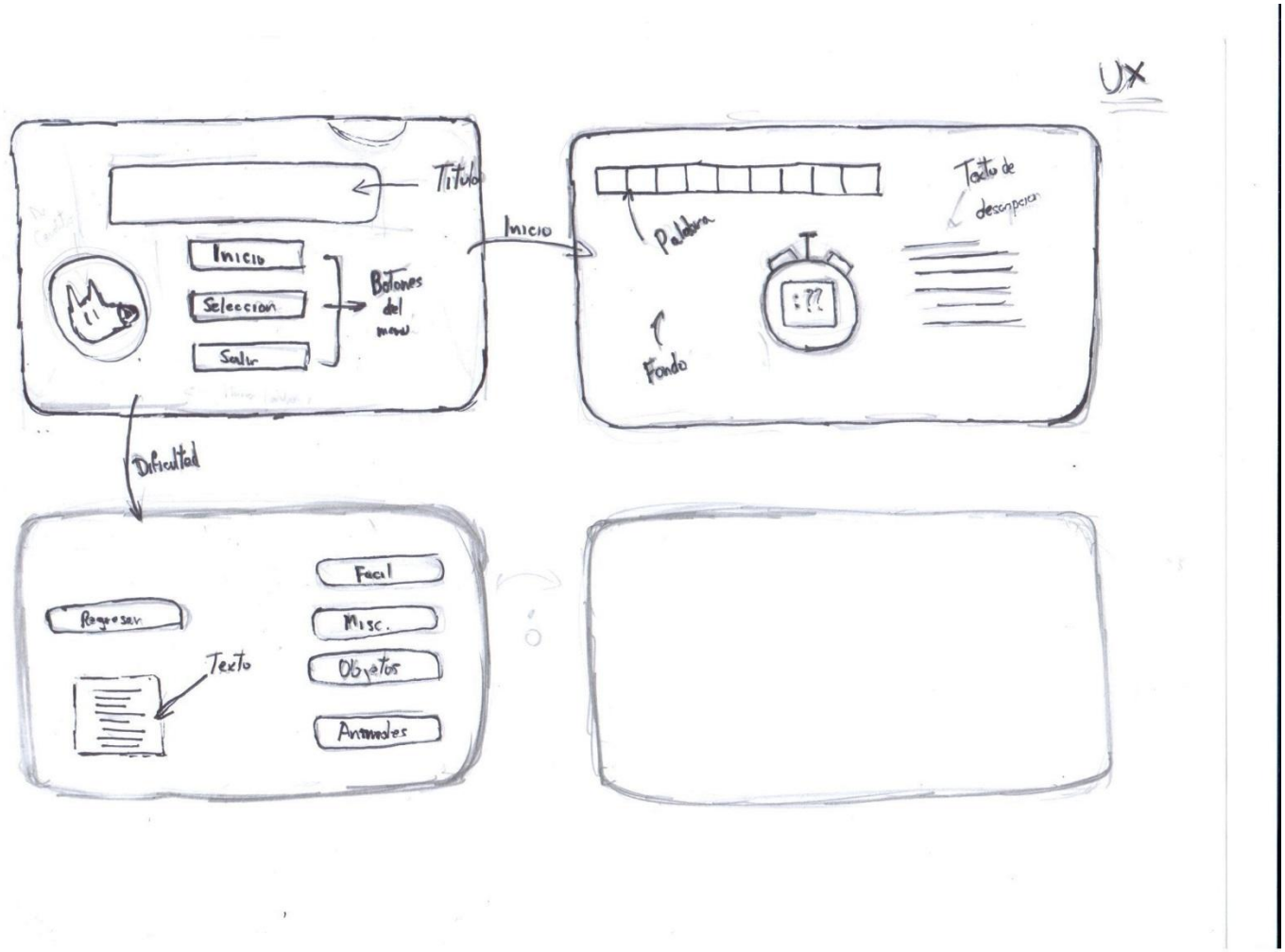


Figura 37: Bosquejos de niveles (creación propia)

Con el diseño de los niveles definidos, se procedió a definir como estarán conformadas las tablas que almacenarían los datos del juego dentro de SQLite (Figura 38). Se definió que la base de datos será conformada por una tabla maestra donde estarán contenidas las clases que tomaran los usuarios, que hacen referencias a una n cantidad de "mazos" de palabras en inglés, junto con sus significados, sonido e imagen.

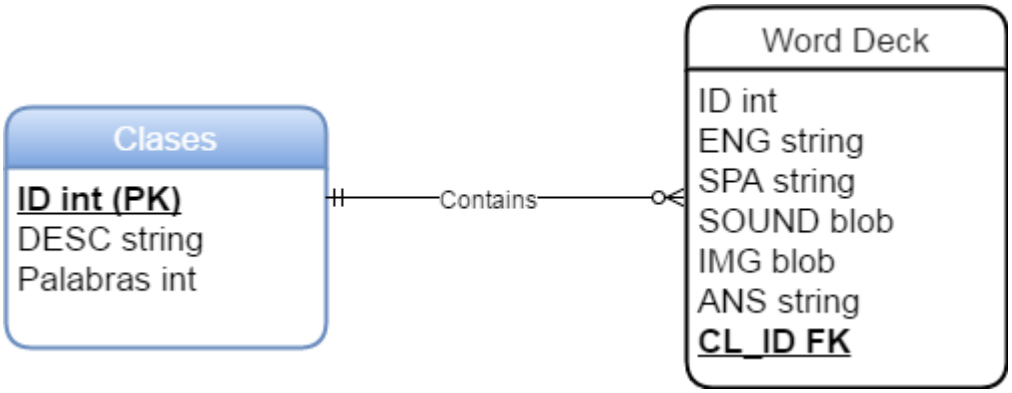


Figura 38: Modelo de Base de Datos escogido (creación propia)

Construcción:

La etapa de construcción involucró el desarrollo del prototipo. En relación a los elementos definidos, esta etapa se divide en la elaboración del aplicativo y luego de la base de datos a acceder para la extracción de la información.

Para la elaboración del aplicativo, se decidió utilizar la plataforma Unity debido a su facilidad de uso, adaptabilidad para las funciones necesarias a programar, su compatibilidad con SQLite y debido a su popularidad como plataforma de desarrollo de videojuegos de diversa gama. Dentro del mismo Unity, aparte de la manipulación de elementos de la misma plataforma, se codifico el comportamiento de los elementos en C#, al igual que la conexión a la base de datos.

Antes de trabajar con el diseño de los mismos niveles, se determinó primero codificar el movimiento del personaje con el cual el jugador podrá interactuar (su avatar en términos

más comunes). Como el género escogido para el juego era de plataforma, se concretó como y cuanto de distancia podía moverse el personaje.

Una vez que el funcionamiento de este comportamiento no mostrara ningún error, se pasó al siguiente paso de programar la lógica de la generación de palabras, culminando en la creación de una clase que controle dicho comportamiento (Figura 39):

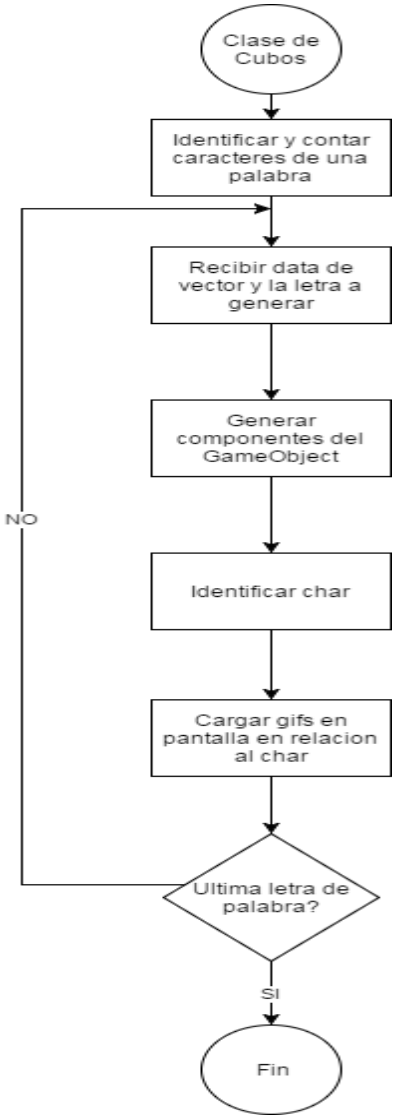


Figura 39: Fragmento de la clase que controla la generación de letras (creación propia)

Con los scripts listos de movimiento y posicionamiento, lo siguiente fue crear los niveles basados en el concept art dentro de la plataforma de Unity. Cada nivel se creó en una escena diferente, y, como cada nivel poseía una escena diferente, con elementos con comportamiento diferentes, se tuvo que crear scripts que los controlen en tiempo real (Figura 40).

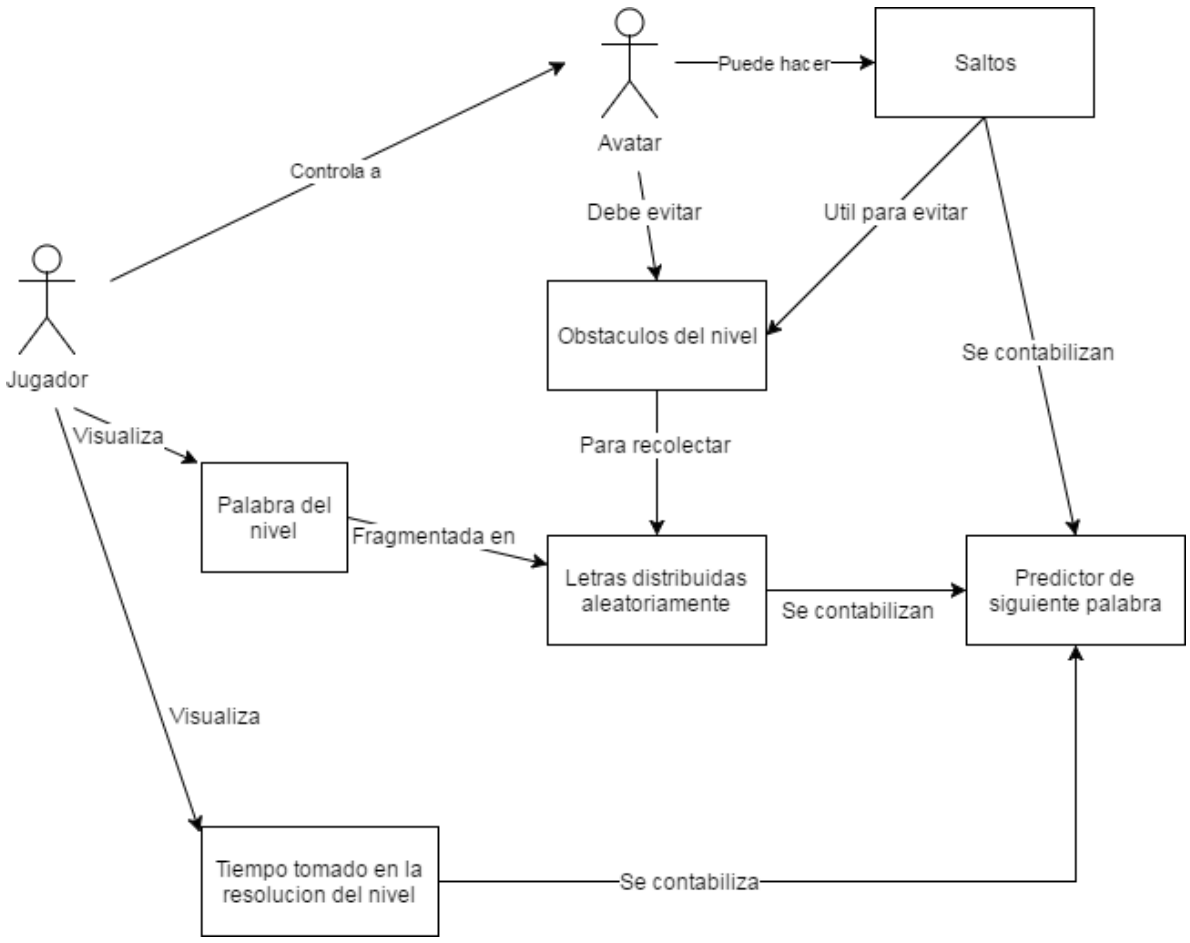


Figura 40: Diagrama de interacción de clases de un nivel (creación propia)

A esta altura, ya se tenía definido los niveles con correcto funcionamiento, con lo que se procedió a crear el selector de niveles para cada escena, además de la creación de un algoritmo que prevenía el acceso a algunos niveles hasta que se logre progresar con uno anterior. A las alturas de este paso, se decidió empezar con el desarrollo de la base de datos.

En el caso de la base de datos, se decidió utilizar SQLite por su característica única de no depender de un servidor en específico para funcionar, además de su facilidad de uso por

medio de un plugin de Mozilla Firefox llamado SQLite Manager. La base de datos fue rellenada con información acerca de las palabras a aprender, su traducción, significado, imagen y audio en relación a su enunciación.

Se creó tablas en similitud a lo previamente establecido, con la tabla “Clases” siendo la principal, las cuales contienen una “n” cantidad de mazos de palabras. Se rellenó la tabla de mazo con 100 palabras, con todos sus campos, para motivos de prueba.

En este punto, se realizó una investigación de si era posible compilar la base de datos en un paquete de apk junto con el juego creado por Unity. Para ello, se investigó y programo un script que utilizaba dlls externos para permitir el acceso de la aplicación a la base de datos cuando esta se pasara a la plataforma de Android.

Con la base de datos en línea, se procedió a crear las escenas de transición, basadas en los concept art de la Figura 37, para que el jugador pueda navegar entre las escenas, además de crear un selector de la clase para cargar las palabras en los niveles durante la sesión.

Pruebas:

En la etapa de pruebas primero se probó el funcionamiento del juego en sí, por medio de un pequeño open alpha en una convención. Con la recopilación de *feedback* en relación al juego, se refino y mejoro la interfaz para la mejor aceptación del videojuego.

Después de ello, se desarrolló una encuesta/examen para determinar el nivel de vocabulario aprendido después de los cambios y comprobar el resultado de la teoría.

Mantenimiento:

La etapa de mantenimiento es en la que se encuentra el prototipo actualmente. Este se encuentra funcional y si surgiese cualquier mejora o inconveniente este será resuelto. Se está refinando y optimizando el contenido del aplicativo para evitar un exceso de “información innecesaria” en relación de los scripts para que el apk resultante sea más ligero para la descarga en Google Play.

CAPITULO IV: RESULTADOS Y ANÁLISIS:

4.1 Resultados:

4.1.1 Predictor:

4.1.1.1 Listado de palabras a utilizar:

Para la prueba del correcto funcionamiento del predictor se utilizó una base de 50 palabras en inglés (Tabla 6). La selección del predictor utiliza 10 de estas palabras aleatoriamente por cada nivel.

BOX	ARCH	IRON	DEBT	PUSH
PIN	OFFER	OVEN	SLEEP	SPOON
BIRD	WATER	DOUBT	FLAME	GLASS
GLOVE	PRICE	RAIL	SCALE	SEED
NEEDLE	FLAG	LAUGH	KNOT	SAND
SCREW	STAR	STEAM	TIME	WAVE
APPLE	BASIN	BRANCH	CANVAS	DISEASE
ENGINE	FEATHER	HARBOR	JOURNEY	LOCK
CRACK	CURTAIN	FORWARD	LETTER	MIND
MOUNTAIN	PAYMENT	MEASURE	RECEIPT	SCISSORS

Tabla 6: Tabla de las palabras del predictor

4.1.1.2 Tabla de reglas:

Para la creación de la tabla de reglas lógicas, las cuales son los resultados que lanza el predictor, se tomó en consideración las cuatro variables de tiempo, aciertos, cantidad de inputs y estado previo (Tabla 7). En relación al tiempo, se consideró un periodo de tiempo a analizar de 30 segundos y la cantidad de input como 15 inputs como máximo. Con esto en consideración, se obtienen **2700** posibles reglas (30 (tiempo en segundos a considerar) x 3 (estados posibles de exactitud) x 15 (cantidad de inputs aceptables) x 2 (estados posibles de resultado)), a comparación que una tabla de 54 posibles reglas. Se realizo dicha extensión debido a la poca cantidad de datos. El siguiente cuadro muestra una abreviación de las reglas.

Tiempo (30 seg)	Exactitud	Input (15)	Estado previo	**RESULTADO**
1	1	1	1	1
1	1	1	0	0
1	1	2	1	1
1	1	2	0	0
1	1	3	1	1
1	1	3	0	0
1	2	1	1	1
1	2	1	0	1
1	2	2	1	1
1	2	2	0	1
1	2	3	1	1
1	2	3	0	1
1	3	1	1	1
1	3	1	0	0
1	3	2	1	1
1	3	2	0	0
1	3	3	1	0
1	3	3	0	0
2	1	1	1	1
2	1	1	0	1
2	1	2	1	1
2	1	2	0	1
2	1	3	1	0
2	1	3	0	0
2	2	1	1	1
2	2	1	0	1
2	2	2	1	1
2	2	2	0	1
2	2	3	1	1
2	2	3	0	1
2	3	1	1	1
2	3	1	0	0
2	3	2	1	1
2	3	2	0	0
2	3	3	1	0
2	3	3	0	0
3	1	1	1	0
3	1	1	0	0
3	1	2	1	0
3	1	2	0	0
3	1	3	1	0
3	1	3	0	0
3	2	1	1	1
3	2	1	0	1
3	2	2	1	1
3	2	2	0	1
3	2	3	1	0
3	2	3	0	0
3	3	1	1	0
3	3	1	0	0
3	3	2	1	0
3	3	2	0	0
3	3	3	1	0
3	3	3	0	0

Tabla 7: Tabla de índice de reglas resumido

El significado de los valores varía en cada fila: (reordenar)

Tiempo: Se consideró a cada segundo como un estado diferente, el cual puede agruparse en tres diferentes conclusiones:

- 1: El usuario completo el nivel antes del tiempo estándar. (1-10 segundos)

- 2: El usuario completo el nivel en el rango esperado de tiempo. (11-20 segundos)

- 3: El usuario completo el nivel excediendo el tiempo estándar. (21-30 segundos)

Exactitud:

- 1: El usuario no completo la palabra al terminar el nivel.

- 2: El usuario terminó el nivel terminando la palabra correctamente. Es decir, la palabra ingresada fue exactamente **igual** que la palabra presentada.

-3: El usuario termino el nivel, pero ingresando una palabra que excedía la cantidad de letras que se presentó.

Para este caso, se va a presentar casos ejemplo de cada cado para una palabra como HOUSE:

- 1: El usuario ingreso HOU u otra combinación de palabra con menor longitud de caracteres que la palabra HOUSE.
- 2: El usuario ingreso HOUSE, es decir, ingreso la palabra exactamente igual que la palabra presentada.
- 3: El usuario ingreso HOUSSE u otra combinación de caracteres con mayor longitud que la palabra house.

Input (Toques en la pantalla):

- 1: La cantidad de inputs necesarios en el nivel es menor de lo esperado (1-5)

- 2: La cantidad de inputs está en el rango esperado. (6-10)

- 3: La cantidad de inputs supera la cantidad predicha. (11-15)

Estado previo:

- 0: El nivel anterior era de **examen** o **prueba** de una palabra previa.

- 1: El nivel anterior era uno de **estudio** de una palabra nueva.

Resultado:

- 0: El siguiente nivel será de **examen**.

- 1: El siguiente nivel será de **estudio**.

Todos estos atributos están relacionados con el algoritmo de elección de la siguiente palabra. Ejemplo: mientras más rápida y exacta sea la respuesta del usuario, incluyendo la cantidad necesaria de inputs en la interfaz, el algoritmo determinará si la siguiente palabra del nivel a presentar al usuario será una de testeo de lo previamente aprendido o una nueva palabra para que este aprenda el nuevo conocimiento.

4.1.1.3 Difusión de la data:

Para este caso en específico, se realizó una difusión de la data para las fases de entrenamiento y testeo, con un 60% de los datos para la fase de entrenamiento (1620 datos), 20% para la validación (540 datos) y 20% para el testeo (540 datos).

Un detalle importante consistió en la **normalización** de todos los datos para la creación de las siguientes matrices en base a la tabla de reglas previa. Además, para entrenar la predictor, se realizó 1000 repeticiones de 100 épocas.

4.1.1.4 Estructura de red

Para conseguir la estructura más eficiente para la red neuronal, se realizaron pruebas con varias posibles estructuras de redes hasta encontrar una estructura que realizara el proceso de cálculo en la menor cantidad de épocas posibles y con el menor nivel de error.

Otro factor a considerar para la definición de la estructura fue la definición del tamaño del *momentum*. Según la cantidad de datos a considerar y la cantidad de tipos diferentes de entradas, se consideraron utilizar tres diferentes *momentums* como prueba. Con esto en mente, se prosiguió a crear tres tablas diferentes (Tablas 8, 9 y 10) para comparar los resultados de error de cada posible estructura (Figura 41) **con el propósito de escoger una para representar la estructura a usar**:

# neuronas	Error promedio testeo	Error promedio entrenamiento	Error promedio validación
2	0.5	0.286266791	0.294454498
3	0.5	0.28178894	0.287750799
4	0.5	0.32438383	0.337166161
5	0.5	0.343387629	0.353635143
6	0.498148148	0.29997338	0.306892937
7	0.5	0.338483501	0.349198256
8	0.412962963	0.275386618	0.280834326
9	0.5	0.345320324	0.354729967
10	0.518518519	0.329053427	0.338654775
11	0.446296296	0.326542293	0.335151528
12	0.438888889	0.338917877	0.34864996

Tabla 8: Tabla de error de la muestra de testeo con *momentum* de 0.0000001 después de entrenamiento

# neuronas	Error promedio testeo	Error promedio entrenamiento	Error promedio validación
2	0.414814815	0.249611793	0.247370758
3	0.301851852	0.244508893	0.243596843
4	0.333333333	0.242590499	0.242639883
5	0.309259259	0.246790874	0.245286232
6	0.281481481	0.241591792	0.238594252
7	0.309259259	0.239617759	0.235378693
8	0.340740741	0.237733351	0.236382176
9	0.331481481	0.243507931	0.241057113
10	0.303703704	0.247307786	0.245094233
11	0.314814815	0.242641482	0.23808116
12	0.301851852	0.236987728	0.23042981

Tabla 9: Tabla de error de la muestra de testeo con *momentum* de 0.00000115 después de entrenamiento

# neuronas	Error mínimo testeo	Error mínimo entrenamiento	Error mínimo validación
2	0.335185185	0.243835779	0.241148744
3	0.412962963	0.253896912	0.254122788
4	0.338888889	0.338888889	0.241614488
5	0.3	0.240756041	0.237852121
6	0.290740741	0.239022089	0.23427144
7	0.290740741	0.239395437	0.234759985
8	0.292592593	0.247614484	0.246517159
9	0.301851852	0.242701852	0.239777707
10	0.312962963	0.246347645	0.243880395
11	0.322222222	0.238505261	0.234573817
12	0.312962963	0.242327239	0.237327039

Tabla 10: Tabla de error de la muestra de testeo con *momentum* de 0.00000125 después de entrenamiento

# neuronas	Error mínimo testeo	Error mínimo entrenamiento	Error mínimo validación
2	0.294444444	0.203470369	0.189679294
3	0.464814815	0.228794697	0.225062443
4	0.388888889	0.221511888	0.212189734
5	0.494444444	0.242742721	0.241503002
6	0.344444444	0.226149727	0.227510772
7	0.316666667	0.202547396	0.211294425
8	0.248148148	0.179229459	0.173402672
9	0.344444444	0.204227476	0.196439176
10	0.353703704	0.198814729	0.195041609
11	0.25	0.166703299	0.157082112
12	0.37037037	0.189386852	0.183953407

Tabla 11: Tabla de error de la muestra de testeo con *momentum* de 0.000625 después de entrenamiento

# neuronas	Error mínimo testeo	Error mínimo entrenamiento	Error mínimo validación
2	0.35	0.209302455	0.209258145
3	0.387037037	0.204955398	0.194794977
4	0.418518519	0.239807451	0.234646573
5	0.268518519	0.1814527	0.166728122
6	0.294444444	0.188916991	0.189090693
7	0.4	0.208561889	0.21190572
8	0.285185185	0.18020447	0.163769904
9	0.3	0.199912464	0.201940618
10	0.144444444	0.136034146	0.124317992
11	0.27962963	0.165307057	0.153367745
12	0.388888889	0.219899787	0.218492949

Tabla 12: Tabla de error de la muestra de testeo con *momentum* de 0.00125 después de entrenamiento

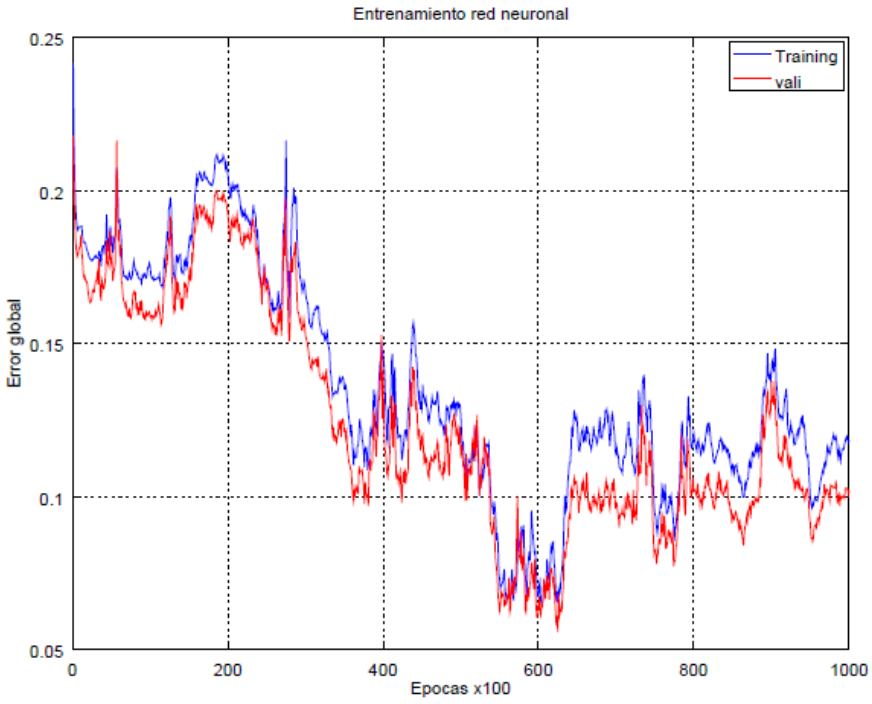


Figura 41: Matriz de error de la red en la fase de entrenamiento seleccionada

Según los resultados obtenidos, se decidió crear una red con 10 neuronas de entrada (+ bias), 7 neuronas en una sola capa oculta (+ bias) y un nodo de salida, similar a la siguiente figura (Figura 42), porque se obtuvo la menor cantidad de errores posibles en cada fase usando dicha estructura; **0.144444444%** en la fase de testeo, **0.136034146%** en la fase de entrenamiento de 2700 y **0.124317992%.**

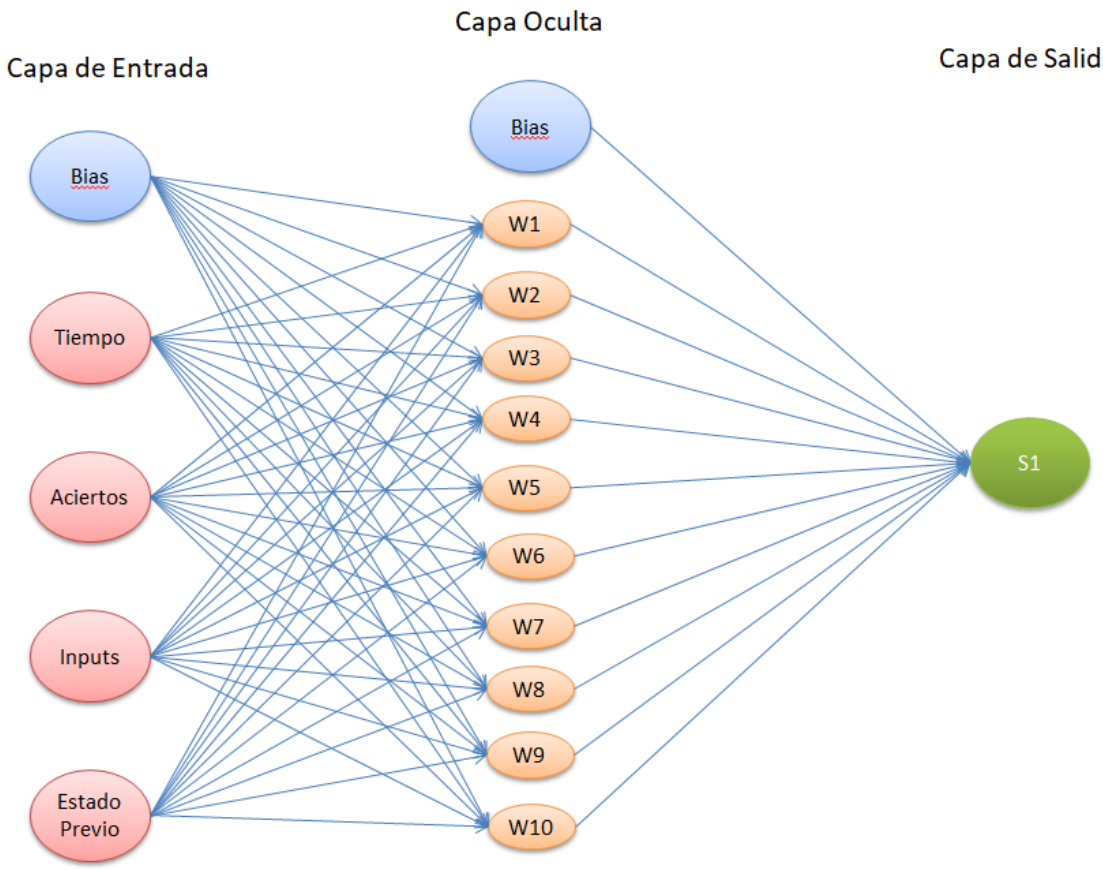


Figura 42: Estructura de la red neuronal (creación propia)

4.1.1.5 Fase de entrenamiento:

Para la fase de entrenamiento, se aplicó funciones de activación **sigmoidales** para las tres capas: entradas, ocultas y salida; con valores ocilantes en -1 y 1, en un periodo de 100 **epocas** repetidas 1000 **veces**, con una variación exponencial por medio de la variable de **momentum** (α). Tomando como referencia a la ecuación 30 para calcular los resultados de cada neurona en relación a su momentum actual:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_j(n)$$

Los 1620 datos de la fase de entrenamiento fueron elegidos aleatoriamente para la formación de la raíz, después de haber normalizado los datos con la siguiente fórmula de normalización:

$$\boldsymbol{Z = \frac{X - \mu}{\sigma}} \qquad Ecuación\ (42)$$

Donde μ es la media de la columna de la entrada y σ la desviación típica de la misma columna de datos. La tabla se encuentra en su totalidad en la parte de anexos.

4.1.1.6 Fase de validación:

En la fase de activación, las funciones de activación para las diferentes capas siguen sido de tipo **sigmoidal**, para un 20% de los datos (540). Se mantuvo el mismo tipo de momentum para la validación.

4.1.1.7 Fase de testeo:

Para la demostración del comportamiento, como es mencionado en la teoría, se probó el rendimiento de la red neuronal ingresados valores previamente no ingresados a la red. Las funciones de activación se mantuvieron iguales (**sigmoidales**) con excepción de la capa de salida, la cual tiene una función de activación de tipo **lineal**. Al ser esta fase la cual nos entrega el resultado final de la red, se tenia que trabajar con un valor entre 0 y 1, con 0.5 como punto medio para determinar cual es el valor position o negativo. La función lineal nos permite calcular el valor resultado entre estos valores.

Demostrado en un diagrama de flujo, se puede comprender el flujo para hallar el resultado como:

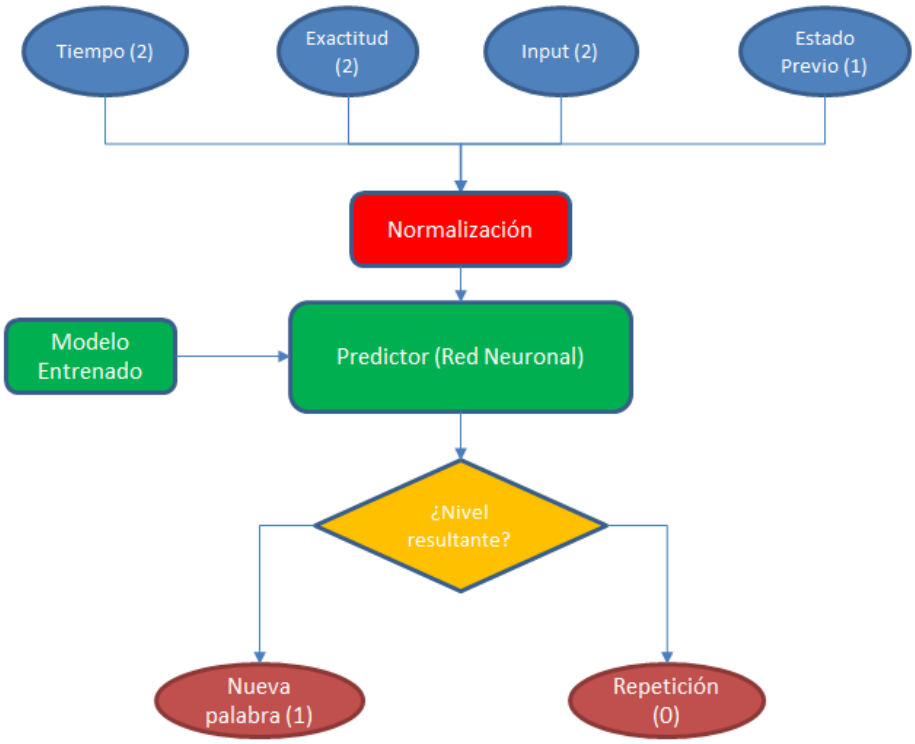


Figura 43: Diagrama de flujo de fase de testeo (creación propia)

En términos matemáticas, se puede representar en la ecuación 43 (Haykin, 2009, p. 196):

$$v_j^{(l)}(n) = \sum_{i=0}^{m_0} w_{ji}^{(l)}(n) y_i^{(l-1)}(n) \qquad Ecuación\ (43)$$

donde $w_{ji}^{(l)}(n)$ es el peso sináptico de la salida de la neurona de resultado del predictor (sea un resultado de nueva palabra o repetición), e $y_i^{(l-1)}(n)$, la salida resultado del predictor. $v_j^{(l)}(n)$ es el resultado total después de la aplicación del peso sobre el resultado de la neurona. Este resultado será redondeado para determinar si se aprenderá una nueva palabra (Resultado = 1) o se repetirá la palabra (Resultado = 0).

4.1.1.8 Definicion y selección del momentum:

Antes de realizar la selección de la cantidad de neuronas necesarias para la red neuronal, se tenía que definir correctamente como se seleccionó los *momentums* propuestos. Se realizó una búsqueda de referencias en papers de tipo A-1, sin embargo, no existía una referencia exacta de la definición del momentum de una red neuronal aplicada a un videojuego. Teniendo en cuenta la ecuación 32, en donde α es equivalente al momentum, y la información relacionado a ello, sabemos que α debe estar entre 0 y 1, similar a como se iniciaron los pesos, y este valor α debe variar **exponencialmente**. Al no tener una referencia clara, se decidió correr experimentos de prueba y error, iniciando aleatoriamente α y variándola exponencialmente para crear las tablas de los posibles errores en relación a las neuronas propuestas para la red.

La razón por la cual se decidió utilizar el aprendizaje por momentum, es decir, por el medio de backpropagation fue por el contexto de la investigación, al ser de trabajar con entradas provenientes de un videojuego. Según el reporte de Sarkar, D (1995), existen varios métodos de calcular los pesos de una red neuronal; sin embargo, en el caso del videojuego propuesto con propósitos educativos, la red debe cambiar para adaptarse a las necesidades y/o capacidades del alumno en cuestión. Por ello, se decido no usar algoritmos normales y entrenar la red para que ajuste sus pesos exponencialmente en relación a los habitos del usuario.

4.2. Población y Muestra

La población del estudio es de un grupo de ciento cuarenta y cuatro (144) estudiantes de rango de 5 a 11 años de edad con acceso a un *smartphone* e Internet.

4.3 Instrumentos de Medida

Se utilizó un prototipo en forma de cuestionario online por medio de Google. En este, se busca evaluar el nivel de vocabulario retenido y/o aprendido después de varias sesiones del uso del aplicativo, tomando en consideración las siguientes variables:

Variables	Indicadores
Efectividad de la aplicación por nivel	Velocidad de solución: tiempo del jugador por nivel / tiempo esperado. Exactitud de solución: Rango de letras de la palabra ingresadas / Rango de letras de la palabra correcta. Solución correcta: Cantidad de letras coleccionadas correctas / Cantidad total de letras de la palabras

Con la efectividad de la aplicación por nivel, se tiene tres indicadores a considerar: la velocidad de solución del nivel, el cual se determina comparando el tiempo actual del jugador con el tiempo esperado para la resolución del nivel, el cual determina que tanto rápido y que tanta confianza el usuario siente para resolver y aprender la palabra del nivel. Mientras más rápido se vuelve el usuario, se puede determinar la efectividad del aprendizaje. Determinando la solución correcta del nivel es otro indicador a considerar, el cual sirve para medir cuantas palabras es capaz de aprender durante la fase de juego de la aplicación.

Finalmente, para determinar si el juego es de agradable uso para el usuario, en relación a las respuestas entregadas por los usuarios en el cuestionario en línea acerca de la satisfacción o comodidad experimentada en el juego, se busca determinar el nivel de satisfacción. Si esta variable es alta, se daría la conclusión que el juego es una herramienta de aprendizaje efectiva,

que no crea aversión para aprender, el cual es un punto a favor en comparación a los otros métodos de enseñanza que existen actualmente en el mundo.

4.3.1 Medida de efectividad del algoritmo:

Un detalle a tomar en cuenta antes de la distribución del aplicativo es la efectividad de la red neuronal para la selección de la palabra. Para ello, se aplicará la siguiente fórmula para calcular el ratio de fallos de la red neuronal:

$$F = \text{casos correctos} / \text{casos totales} * 100 \qquad Ecuación\ (44)$$

Esta matriz muestra la cantidad de instancias erradas del algoritmo de selección, con el propósito general de identificar el ratio correcto de falla para así tener un mejor panorama de confiabilidad del programa cuando se empiece a realizar las pruebas respectivas del caso.

4.3.2 Ponderación de la respuesta en encuestas de satisfacción:

Un índice de satisfacción global del cliente puede ser, simplemente, el promedio de puntuaciones medias otorgadas por los clientes a los diferentes aspectos del servicio: dadas las dimensiones o aspectos a, b, c ... n, las puntuaciones medias de los clientes a dichas dimensiones son μa, μb y μc ... μn , y el índice de satisfacción global entonces es:

$$\mathbf{ISG} = (\mu a + \mu b + \mu c + ... + \mu n)/n, \qquad Ecuación\ (45)$$

…que es una cifra que sintetiza mucho mejor que la distribución de porcentajes de las respuestas al cuestionario. Dicho en sí, esta medida se utiliza para medir el nivel de satisfacción general en algunas respuestas brindadas en las encuestas.

4.4 SCRUM

4.4.1 Resultados de SCRUM:

De lo definido en la teoría de SCRUM y los objetivos a cumplir para la creación del producto, se definió los principales Sprints (Figuras 43, 44 y 45):

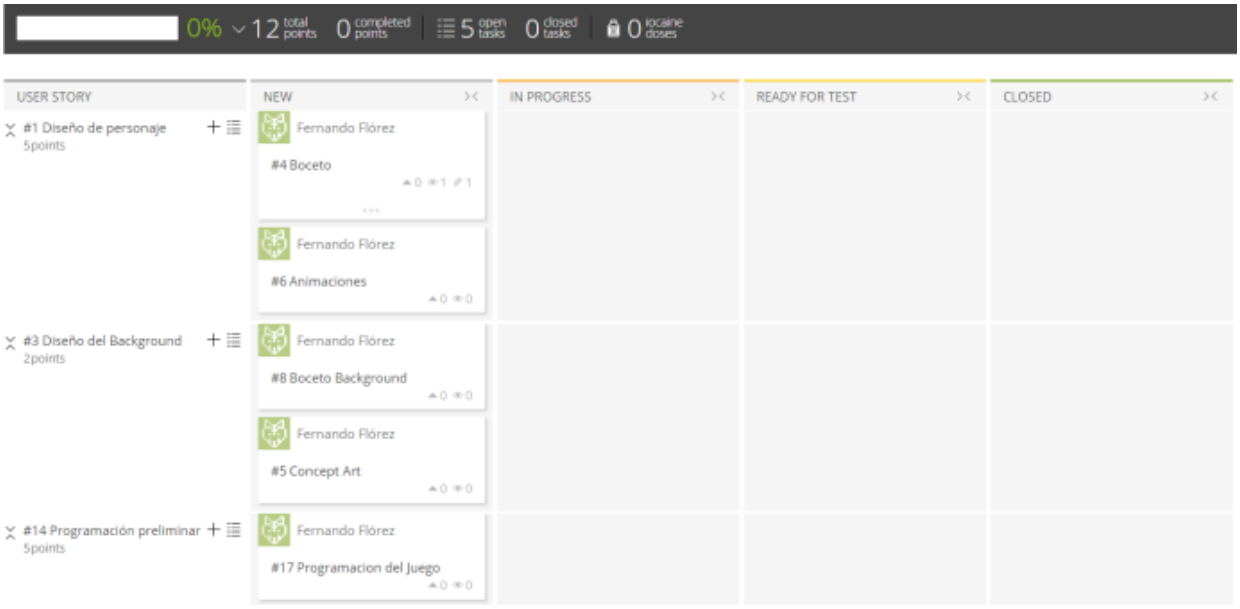


Figura 44: SPRINT de la Creación del Prototipo

Fuente: Taiga.io (creación propia)

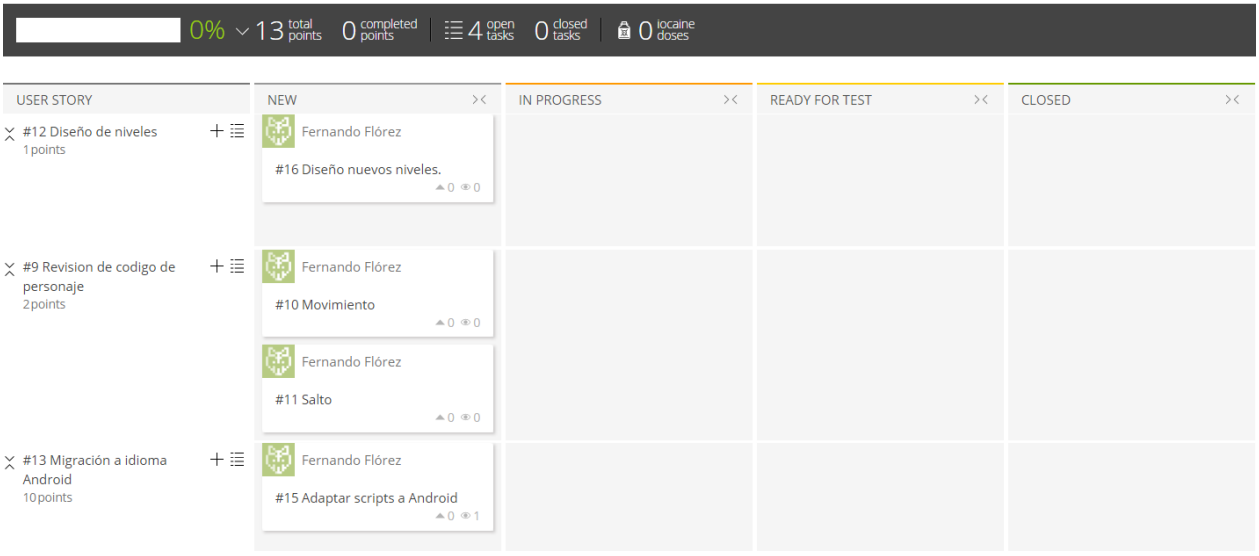


Figura 45: SPRINT relacionado al tema de programación

Fuente: Taiga.io (creación propia)

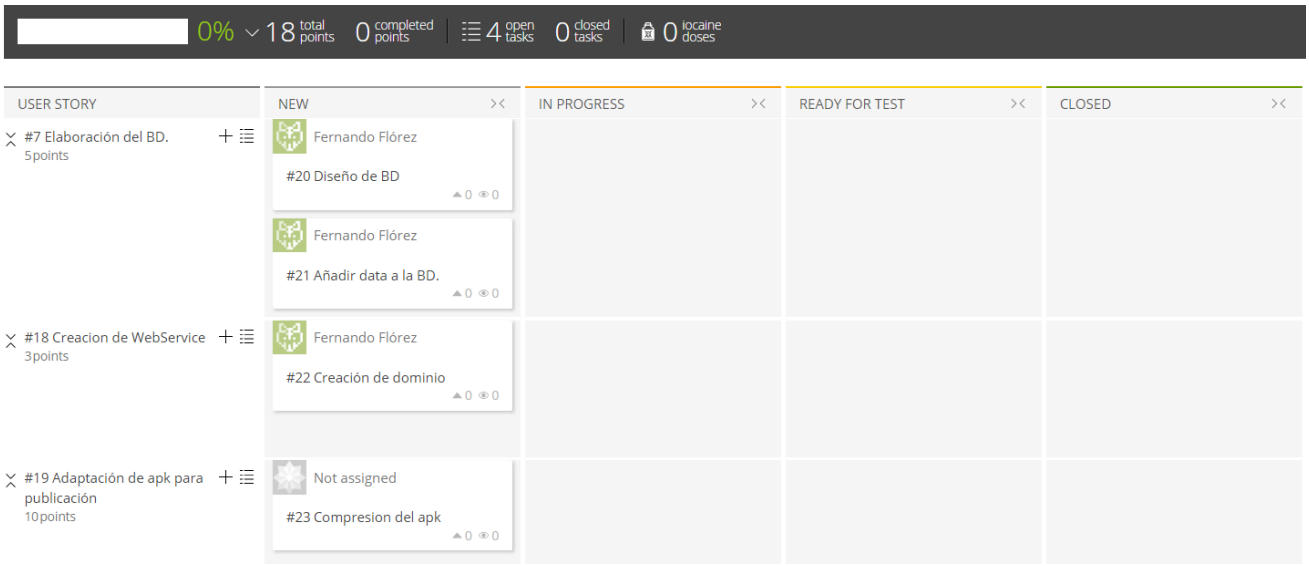


Figura 46: SPRINT BD, WebService, Apk

Fuente: Taiga.io (creación propia)

Desde los meses de Noviembre del 2016 a fines de Enero de este año (2017), se ha realizado revisiones de los avances del proyecto, mejorando y discutiendo las mejoras y

correcciones necesarias para obtener la aceptación del público objetivo. De lo denotado por la teoría, con el uso de la misma, se logró recortar y agilizar el proceso de trabajo.

Durante las semanas de Diciembre, se culminó con proceso del SPRINT del prototipo (en relación a la parte que no tenía relación con la programación) y se pasó a entrar a definir y programar los comportamientos necesarios para probar la factibilidad de la hipótesis (Figura 46)

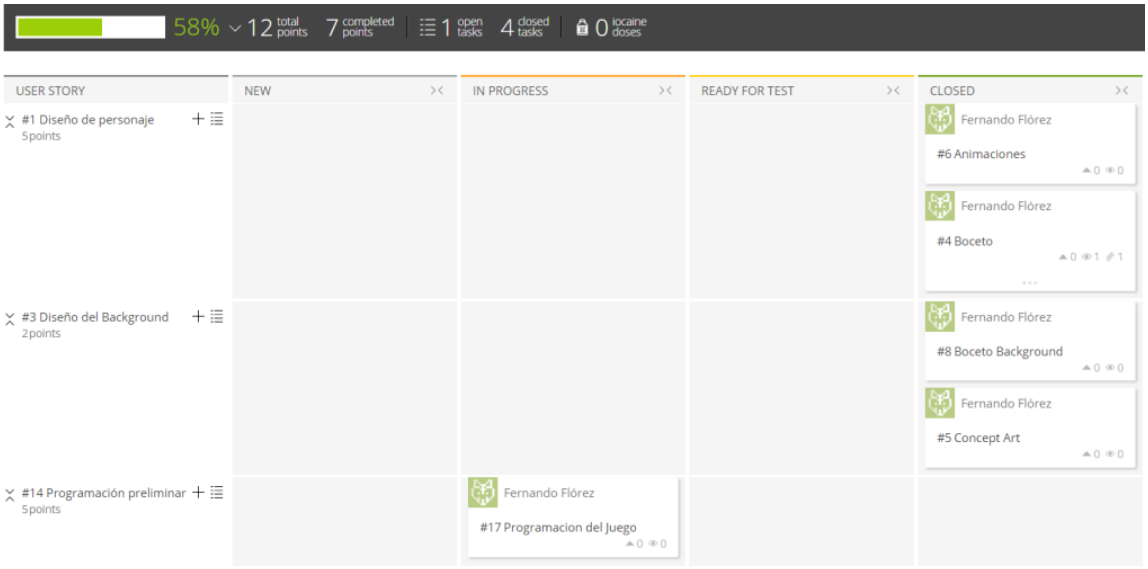


Figura 47: SPRINT de la Creación del Prototipo (Diciembre)

Fuente: Taiga.io (creación propia)

A inicios de Enero, se finalizó la programación preliminar y se prosiguió con la programación a fondo. En paralelo, se comenzó a definir lo que debía contener la base de datos (Figuras 47, 48 y 49):

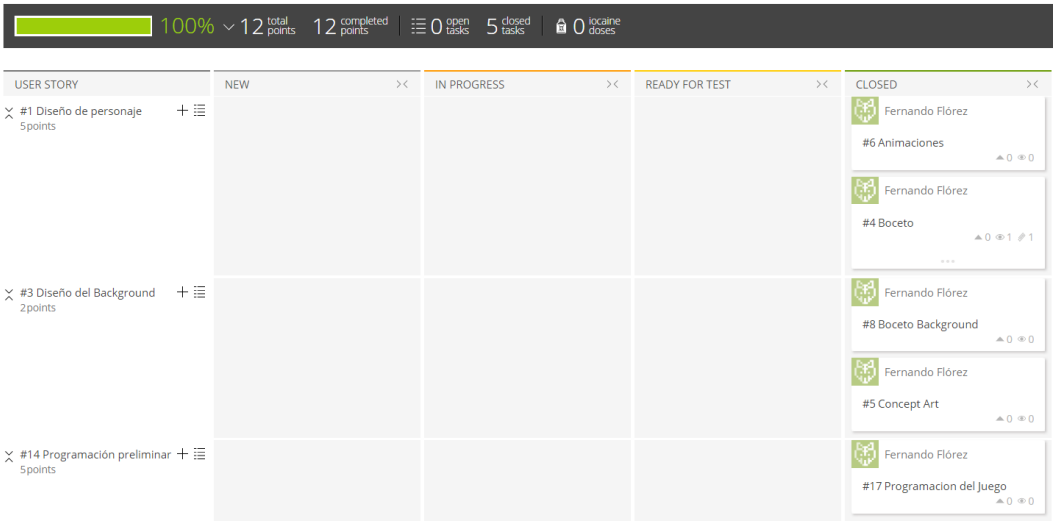


Figura 48: SPRINT de la Creación del Prototipo (Inicios de Enero)

Fuente: Taiga.io (creación propia)

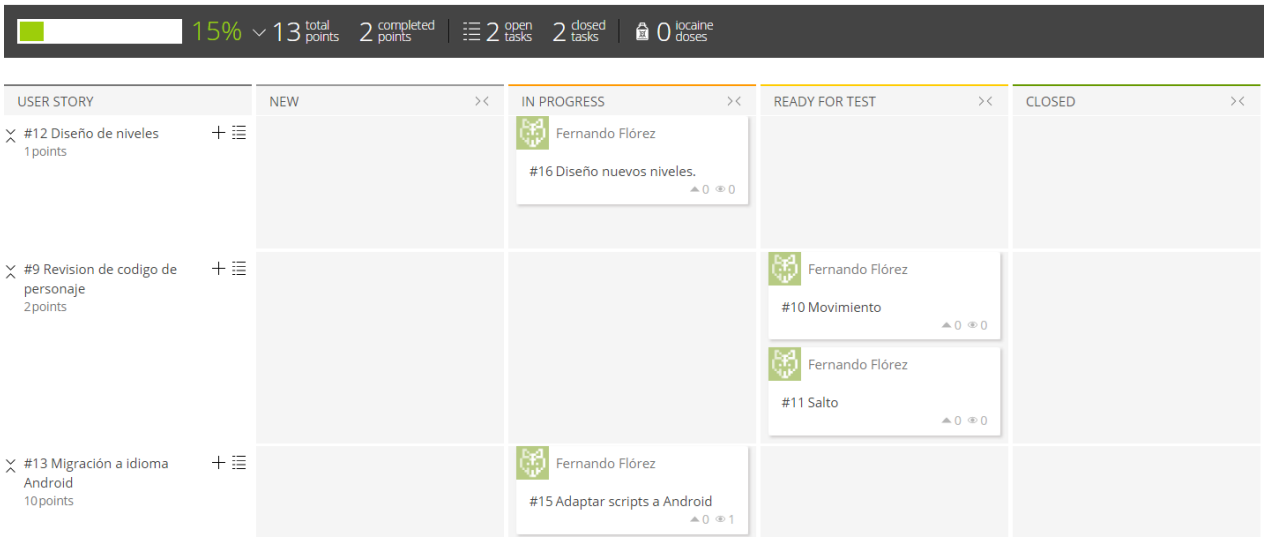


Figura 49: SPRINT relacionado al tema de programación (Inicios de Enero)

Fuente: Taiga.io (creación propia)

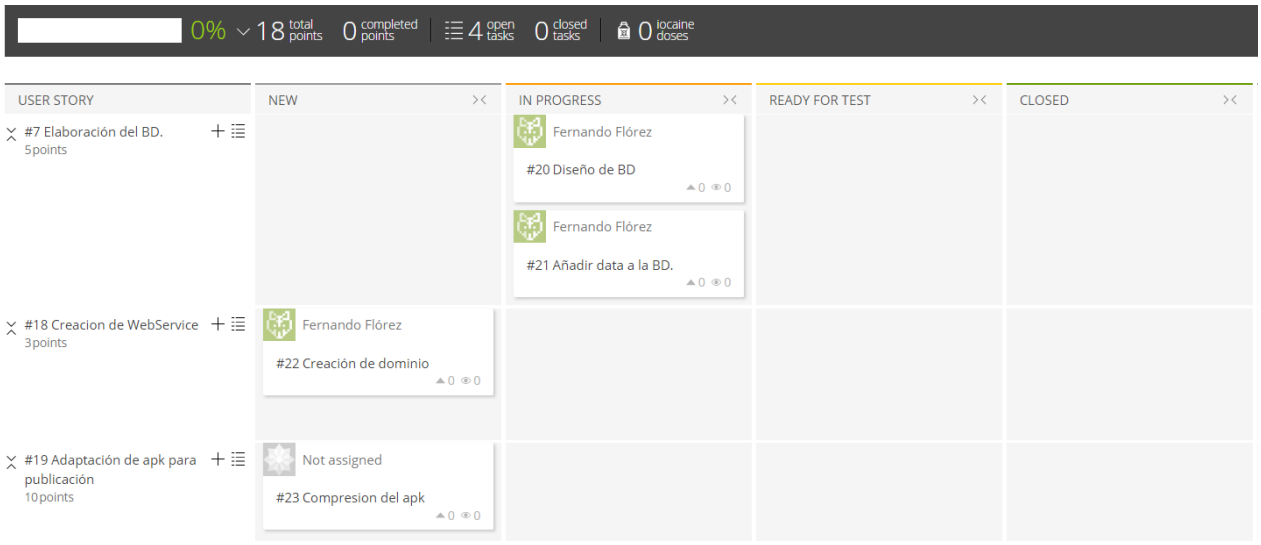


Figura 50: SPRINT BD, WebService, Apk (Inicios de Enero)

Fuente: Taiga.io (creación propia)

Términos como el uso de base de datos, uso de webservices y redes neuronales fueron temas discutidos en las reuniones y aceptadas para ser programadas en el proyecto (Figuras 50 y 51):

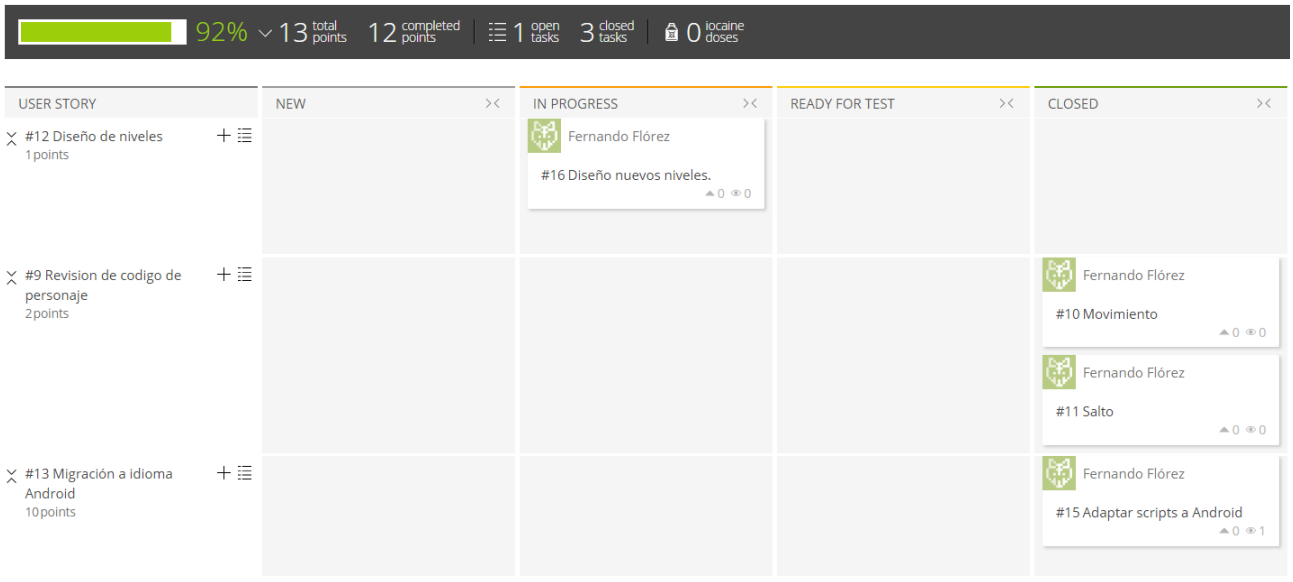


Figura 51: SPRINT relacionado al tema de programación (Fines de Enero)

Fuente: Taiga.io (creación propia)

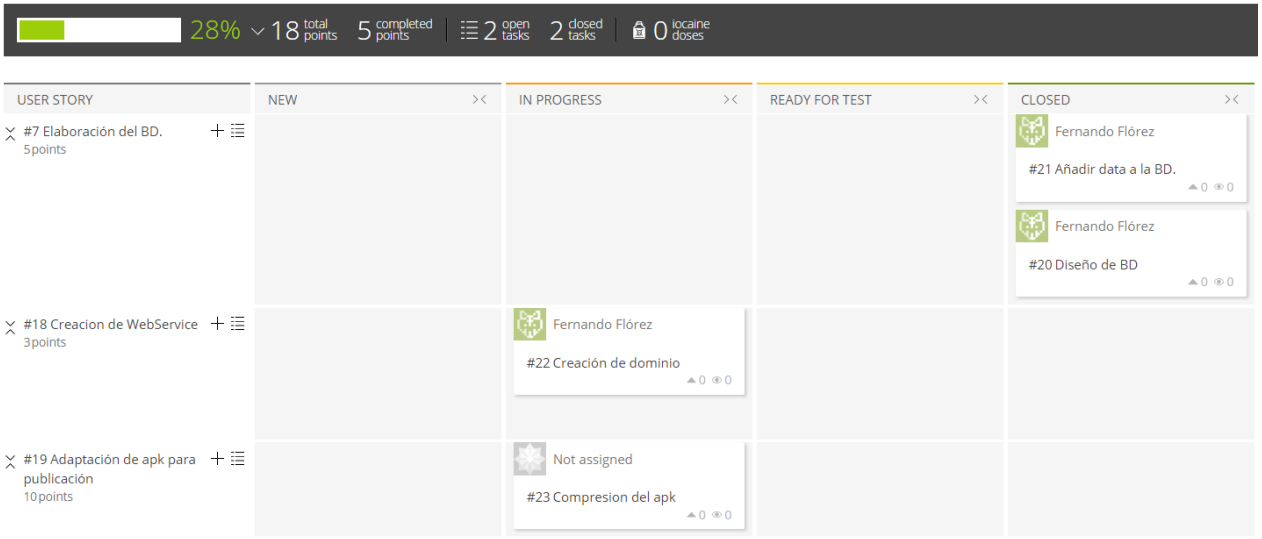


Figura 52: SPRINT BD, WebService, Apk (Fines de Enero)

Fuente: Taiga.io (creación propia)

Finalmente, una vez probada la funcionalidad del código para su comunicación con base de datos y con urls de prueba, se empezó el *stretch* final de la elaboración de la base de datos y la elaboración del webservice, para facilitar la recolección de la información necesaria (Figura 52):

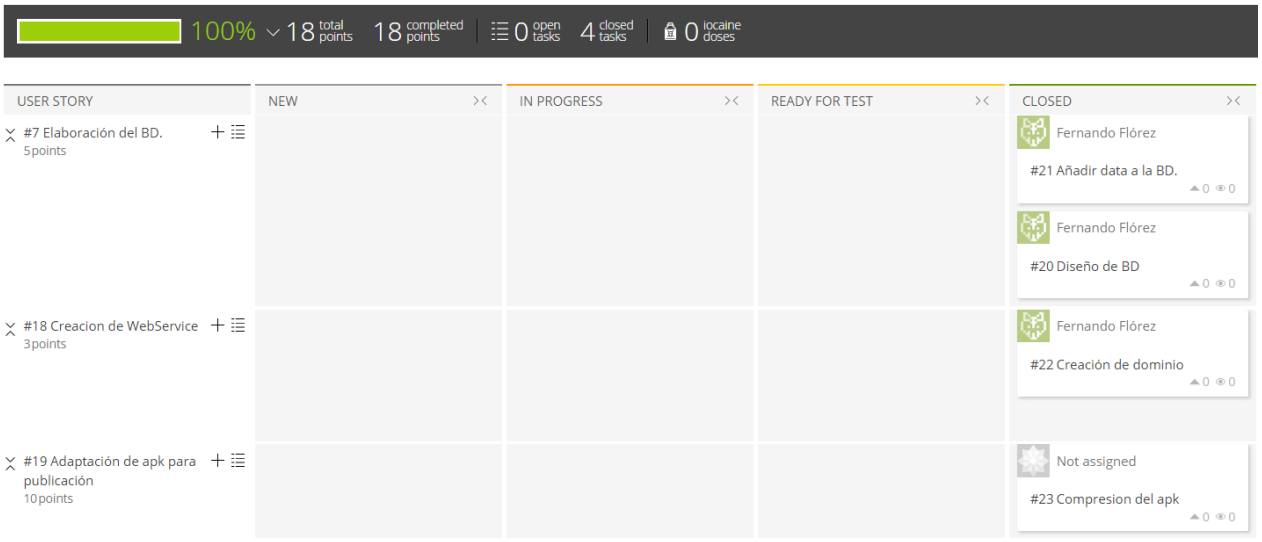


Figura 53: SPRINT BD, WebService, Apk (Febrero)

Fuente: Taiga.io (creación propia)

En conclusión, por medio de SCRUM, se logró agilizar el avance del proyecto, al mismo tiempo que se reforzaron pasos anteriores para que estén en línea con la dirección del proyecto una vez que se definieron términos como el uso de base de datos y aplicación de redes neuronales.

4.4.2 Resultados de las encuestas:

En relación a lo mencionado anteriormente, tenemos tres hipótesis a probar:

H1: Existe una relación directa entre la inclusión de dificultad de los niveles con el aprendizaje del usuario.

H2: Existe una relación directa entre la inclusión de pistas audiovisuales en el videojuego con el aprendizaje del usuario.

H3: Existe una relación directa entre la cantidad de palabras acertadas por el usuario y el impacto de la enseñanza del juego.

Para la validación de las mismas, se planteó una encuesta virtual a una cantidad establecida de 144 participantes, entre niños y niñas. Para la primera hipótesis, se tomó en cuenta los siguientes resultados (con un objetivo de aceptación de mínimo 80%) (Figuras 52 y 53) y se usó **la ecuación 44**, con 400 siendo el valor del 100% multiplicado por las motivaciones positivas (4):

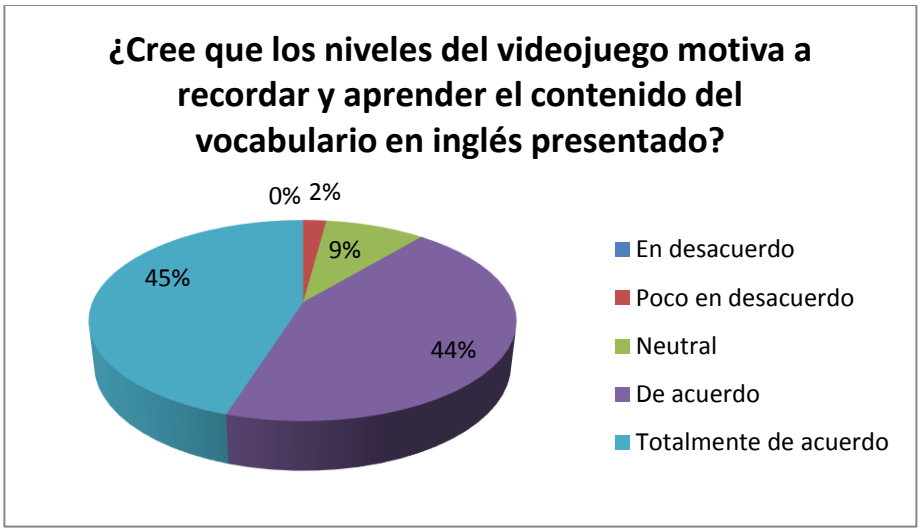


Figura 54: Acerca de la relación de dificultad de niveles y aprendizaje (creación propia)

ISG = (45*4 + 44*3 + 9*2 + 2*1 +0*0)/400 = 0.83

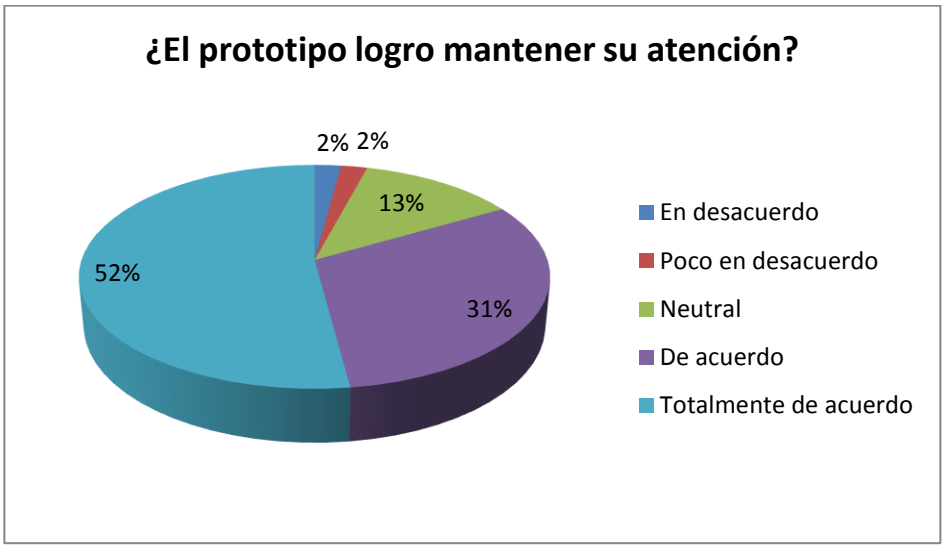


Figura 55: Acerca de la captación de atención (creación propia)

ISG = (52*4 + 31*3 + 13*2 + 2*1 +2*0)/400 = 0.8225

Según los resultados obtenidos, se da por demostrar que un porcentaje superior al 80% en promedio de todas las respuestas han sido correctas. La conclusión resulta ser que, **SI** existe una relación directa entre la inclusión de elementos audiovisuales en el videojuego y el aprendizaje logrado por el usuario, por lo cual no se rechaza la hipótesis H1.

Ámbito visual

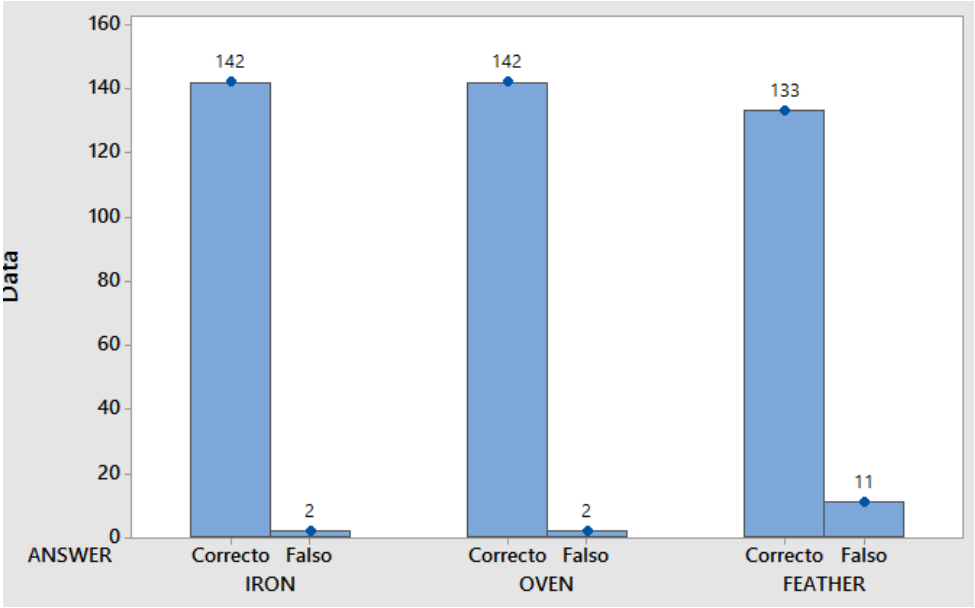


Figura 56: Barras en relación a los resultados del ámbito visual (creación propia)

Ámbito auditivo

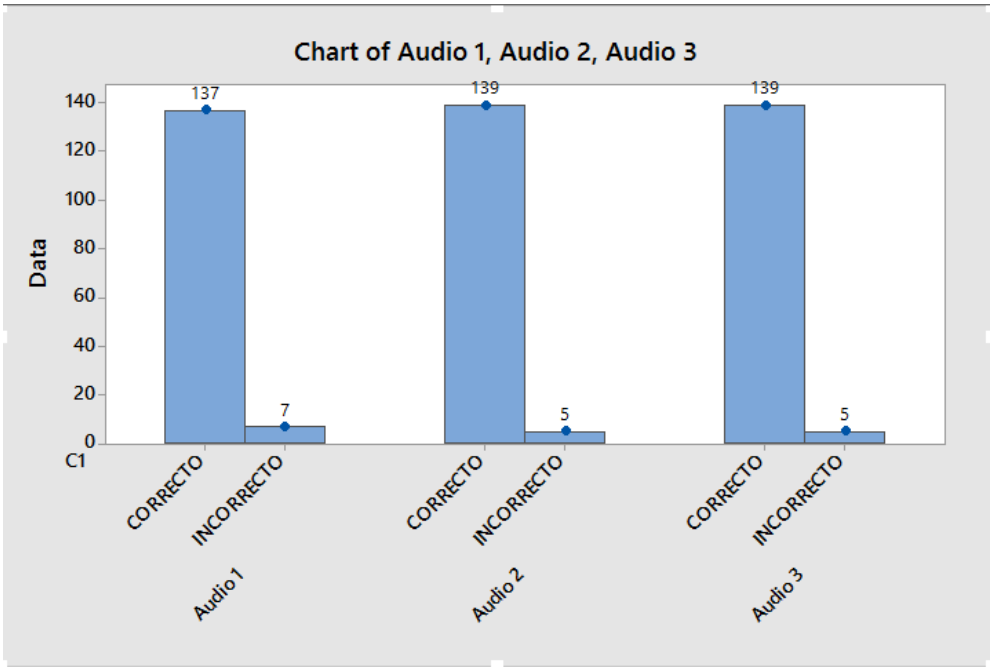


Figura 57: Barras en relación a los resultados del ámbito auditivo (creación propia)

En relación a la hipótesis H2, los resultados de las pruebas visuales y audiovisuales, demostrados en las Figuras 55 y 56, determinan con certeza que más de un 90% de instancias los usuarios lograron asociar el audio e imágenes con el significado correspondiente, por lo cual la hipótesis H2 no se rechaza, debido a la relación positiva entre el contenido audiovisual y el aprendizaje del usuario.

Finalmente, para la hipótesis H3, se tomaron en cuenta los siguientes resultados (Figura 57):

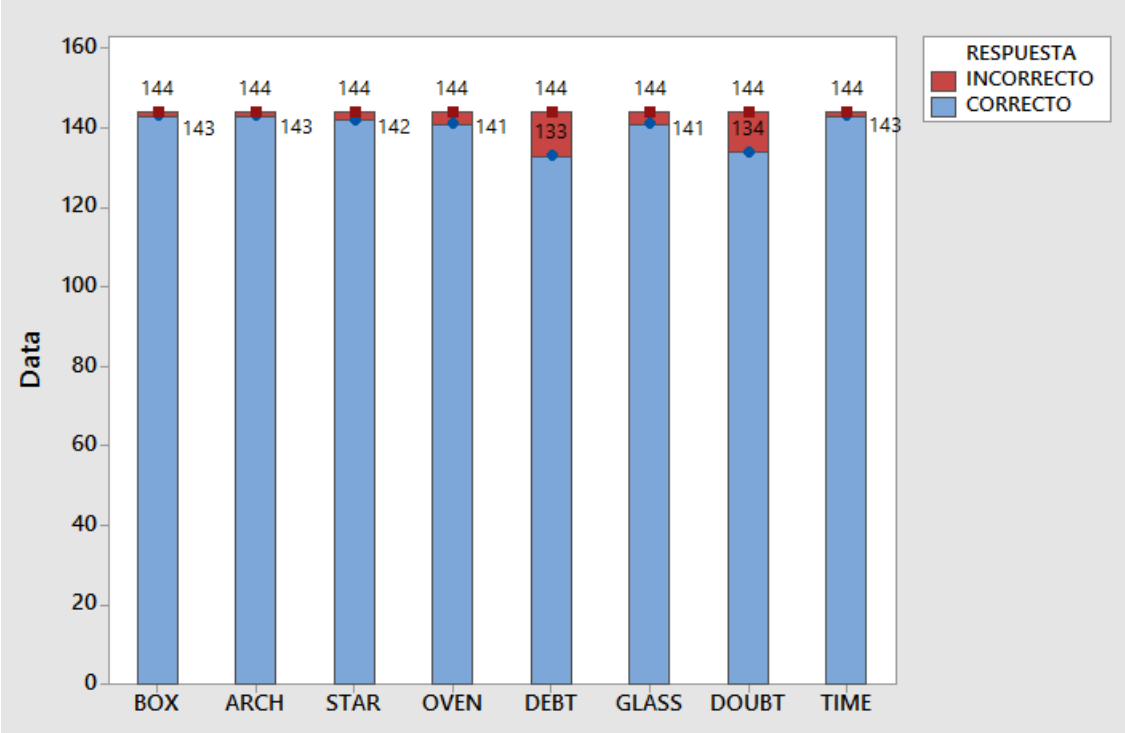
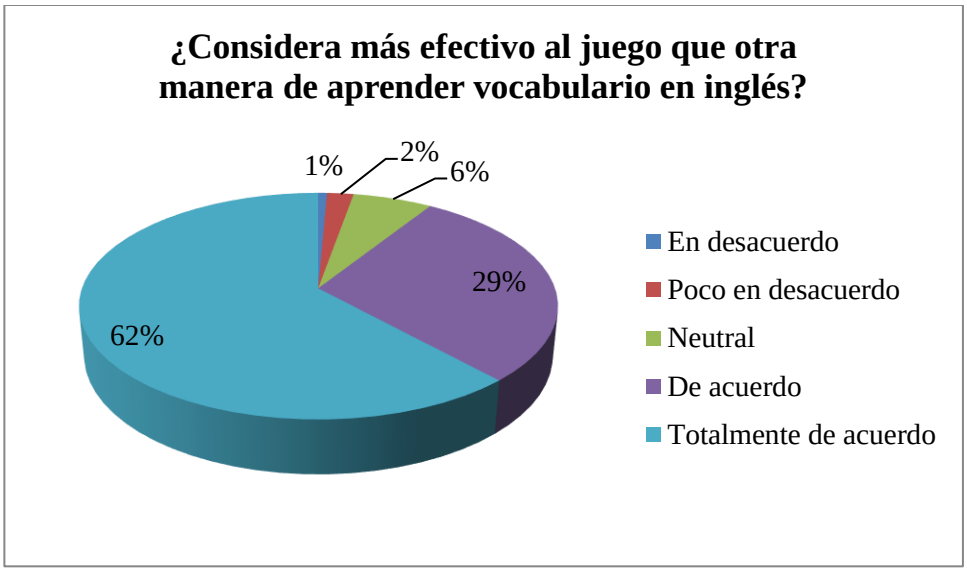
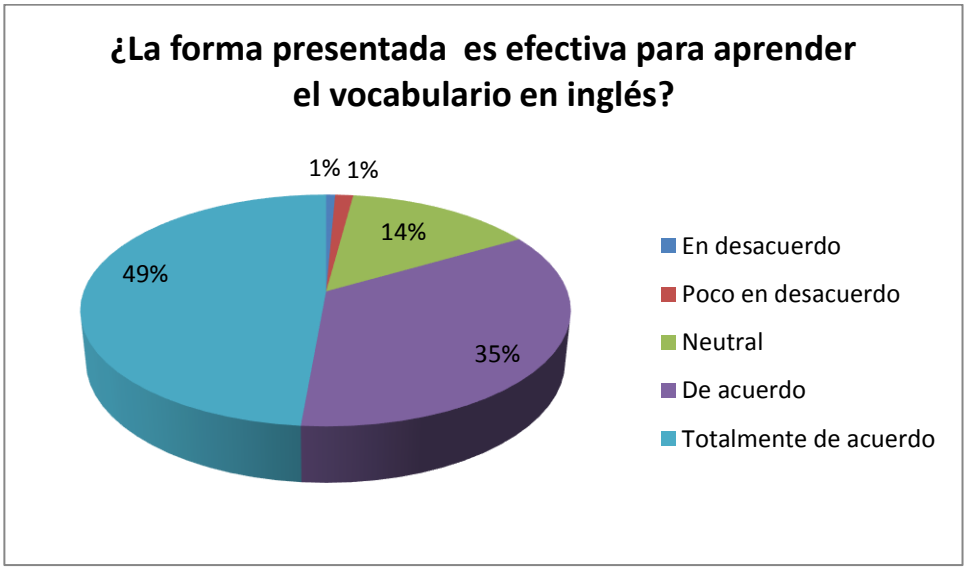


Figura 58: Barras en relación a los resultados del checkbox en la encuesta (creación propia)



ISG = (62*4 + 29*3 + 6*2 + 2*1 +1*0)/400 = 0.8725



ISG = (49*4 + 35*3 + 14*2 + 1*1 +1*0)/400 = 0.825

Figura 59 y 60: Gráficas que muestra la opinión del encuestado acerca de la efectividad de los videojuegos en la enseñanza (creación propia)

Según las figuras 56 y 57, las respuestas presentadas en la encuesta muestran un porcentaje promedio mayor al 90% de aciertos después de jugar el videojuego, proporcionando gran confianza en la prueba positiva de la tercera hipótesis. Adicionalmente

en relación a los resultados en las Figuras 58 y 59, se puede determinar con un porcentaje superior al 80% que los videojuegos son una manera efectiva para aprender vocabulario en inglés. Con estos resultados, se puede afirmar que, **SI** existe una relación directa entre la cantidad de palabras acertadas por el usuario y el impacto de la enseñanza del juego, es decir, se acepta la hipótesis H3.

4.4.3 Resultados de las pruebas de variables (Prueba Z de 1 población):

En la parte 3.3 de esta tesis, se determinó tres variables a analizar para determinar el nivel de vocabulario retenido por los participantes: Dichas variables eran:

- Velocidad de solución: Tiempo del jugador vs tiempo esperado.
- Exactitud de solución: Rango de error real vs rango de error determinado.
- Solución correcta: Cantidad total de palabras resueltas correctamente vs total de palabras encontradas por los participantes.

  Un detalle importante a considerar es la manera como se definió por código una palabra correcta. Usando la interfaz de juego y considerando cada letra de la palabra como un objeto a considerar, se definió una palabra como una lista de caracteres. Una palabra se considera correcta cuando la palabra ingresada por el usuario mediante la interfaz del juego es igual a la que se encuentra en la base de datos generada por el sistema. Es decir, se comparan los caracteres ingresados en contra de los caracteres que conforman a la palabra correcta.

Para encontrar estas incógnitas, aparte de la distribución de encuestas, se midió el desempeño del usuario por medio del uso de web services dentro del videojuego el cual almacenó la información necesaria para probar que el videojuego era efectivo para la retención del vocabulario en inglés del usuario (Figuras 60 y 61):

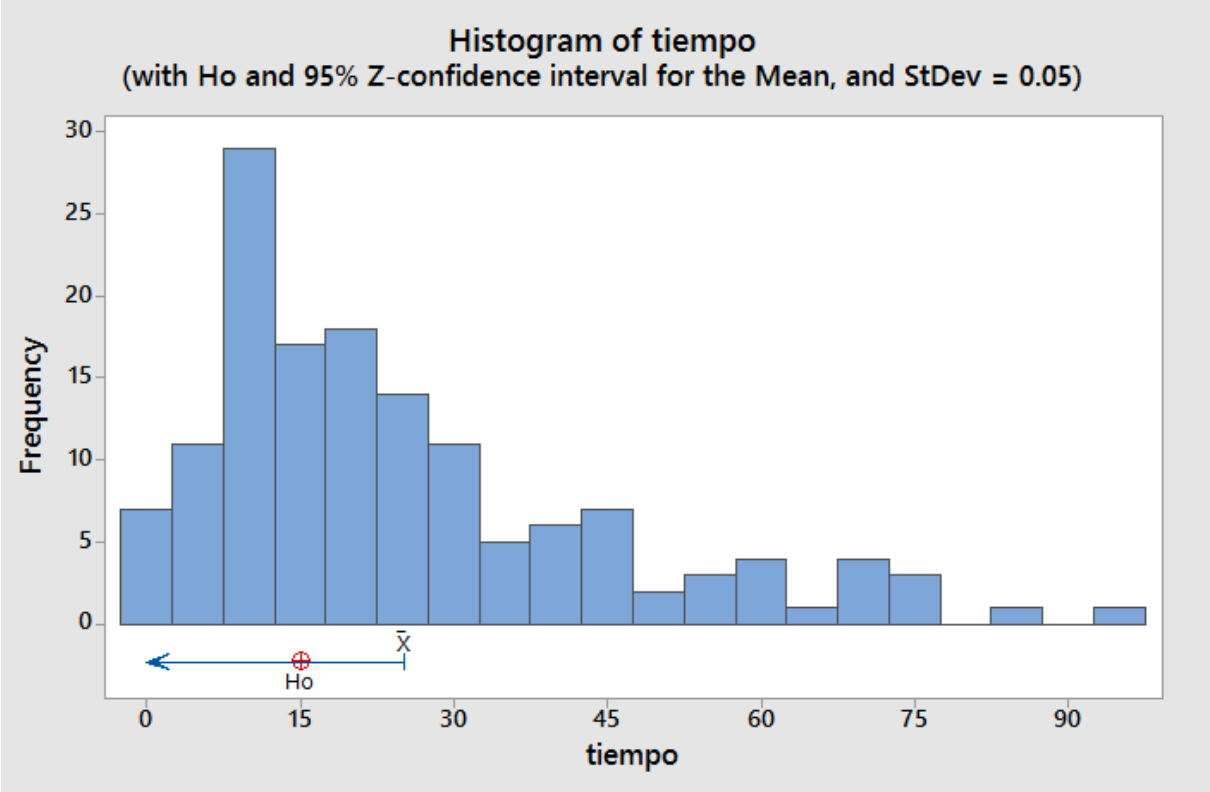


Figura 61: Prueba Z de 1 población del tiempo necesario para completar un nivel (creación propia)

3/28/2017 8:55:58 PM

One-Sample Z: tiempo

```
Test of μ = 15 vs < 15
The assumed standard deviation = 0.05

Variable    N     Mean    StDev  SE Mean  95% Upper Bound        Z      P
tiempo    144  25.1736  20.0295   0.0042          25.1805  2441.67  1.000
```

Figura 62: Datos de la prueba Z de la variable "tiempo" (creación propia)

Usando un grado de confianza del 95% de la muestra se buscó comprobar:

Hao: El tiempo promedio de resolución de los niveles fue más bajo que la media estimada.

Ha1: El tiempo promedio de resolución fue más grande que la media estimada.

Y se da la conclusión de *rechazar* Hao debido a los resultados. El motivo principal por el cual se llegó a rechazar la conclusión Hao se debió a la dificultad en el control de la aplicación para el usuario. Los controles del aplicativo resultaron ser muy difíciles de entender al inicio a los usuarios, y eso se tradujo en un ratio de demora más alta de lo normal para la resolución de los niveles. Pruebas consecuentes, sin embargo, se mostraron más positivas. (Figuras 61 y 62):

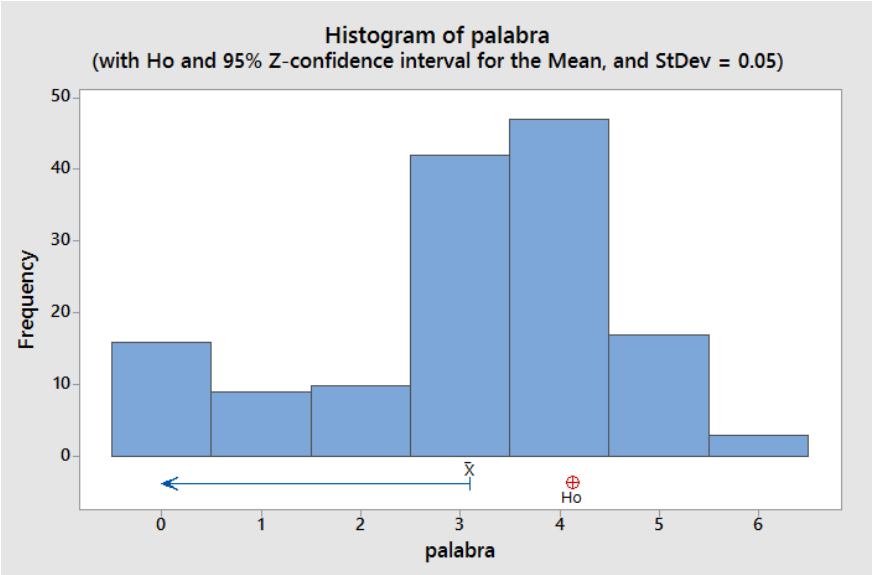


Figura 63: Prueba Z de 1 población del exactitud y solución correcta de la palabra (creación propia)

(En relación al rango de error)

One-Sample Z: palabra

```
Test of μ = 4.13 vs < 4.13
The assumed standard deviation = 0.05

Variable    N     Mean    StDev  SE Mean  95% Upper Bound        Z      P
palabra   144  3.09722  1.52058  0.00417          3.10408  -247.87  0.000
```

Figura 64: Datos de la prueba Z en relación a las palabras correctas. (creación propia)

En relación a la información presente, se tiene las hipótesis:

Hb0: El rango de error de las palabras de los niveles ha sido menor al rango de error establecido.

Hb1: El rango de error de los niveles fue superior al rango de error establecido.

El cual, en relación a los resultados presentados en la gráfica de prueba de Z, se llega a aceptar Hbo. En relación a los resultados, se logró identificar un alto ratio de aciertos de las palabras presentadas. Los usuarios lograron identificar y relacionar las imágenes y sonido con los resultados correctos.

4.4.4 Resultados de las redes neuronales (predictor):

4.4.4.1 Matriz de pesos:

Al inicializar la matriz de pesos, se definieron todos los pesos de manera aleatoria, entre 0 y 1, la razón detrás de ello se debe a que se utilizó la inicialización de distribución uniforme (Glorot, X., Bengio. Y, s.f.). La aplicación de dicha inicialización se expresa por la fórmula:

$$W \sim U\left[-\frac{1}{\sqrt{n_{in}}}, \frac{1}{\sqrt{n_{in}}}\right] \qquad Ecuación\ (46)$$

Se colocó un bias de 0.5 a los nodos correspondientes. Esta son las tablas de pesos antes del entrenamiento (Tablas 12 y 13):

Bias	Tiempo	Aciertos	Input	Estado Previo
0.5	0.70090634	0.43536075	0.71902706	0.82180944
0.5	0.0839622	0.45469392	0.27008848	0.35904447
0.5	0.48894191	0.81088633	0.35330163	0.55540995
0.5	0.27123311	0.00920126	0.8803578	0.10989635
0.5	0.21212149	0.03016229	0.14218519	0.01617727
0.5	0.7468583	0.57709615	0.49164004	0.3407587
0.5	0.0441941	0.33338662	0.35656094	0.32912323
0.5	0.04354605	0.47682864	0.10597549	0.53657894
0.5	0.07003808	0.39997315	0.68048018	0.8675439
0.5	0.62763096	0.75667228	0.83624995	0.38493465

Tabla 13: Tabla de pesos iniciales de la capa de entrada antes del entrenamiento

Bias	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
0.5	0.2909	0.5138	0.5171	0.0010	0.0466	0.4821	0.1019	0.0708	0.1220	0.5445

Tabla 14: Tabla de pesos iniciales de la capa de salida antes del entrenamiento

Una vez aplicado el entrenamiento a la red, los pesos fueron reajustados utilizando Z, es decir, la fórmula de normalización. (Tablas 14 y 15):

Bias	Tiempo	Aciertos	Input	Estado Previo
4.39433451	1.94293382	-0.45046553	0.50137051	1.54513873
5.82092156	-0.0724573	0.02547957	0.26877236	0.49512119
8.15253385	0.25701135	0.38283594	-0.05040708	1.09026286
4.72302443	1.80769833	0.55023533	1.37235548	-0.7619582
-4.31258332	0.5601017	-3.80821221	1.55134656	-1.39638078
7.33085271	0.67376261	-0.07172188	0.51946761	-0.39611428
8.67864782	-0.02119245	0.69967333	-0.17728187	1.03666014
-4.50260089	-0.94119539	1.27830444	0.88109183	-1.81455854
-2.57930416	-0.34538483	0.40762249	1.28107478	2.32154372
-1.99916667	2.74064156	1.75792046	-0.66176815	-1.33437484

Tabla 15: Tabla de pesos de la capa de entrada después de la fase de entrenamiento

Bias	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
-7.913	19.691	0.57295	5.8655	2.8258	-13.955	-12.320	5.3106	-11.613	-14.775	-13.521

Tabla 16: Tabla de pesos de la capa oculta después de la fase de entrenamiento

4.4.4.2 Resultados de prueba

Durante la fase de testeo, se comprobó la efectividad del predictor, cuya exactitud de predicción llega a superar el 95%, aproximándose a la certeza absoluta..

4.5 Resultados de la matriz de confusión

Para verificar la veracidad y exactitud de la red neuronal empleada, se realizó una prueba con 135 usuarios, probando un prototipo de la aplicación. El prototipo del videojuego tiene la capacidad de recopilar la información de cada sesión de juego de los participantes, lo cual sirvió para la determinación del buen funcionamiento del predictor de niveles. La información recopilada se muestra a continuación (Tabla 16):

	PREDICTED YES	PREDICTED NO
ACTUAL NO	FP = 1	TN= 60
ACTUAL YES	TP= 68	FN= 6
	69	66

Tabla 17: Tabla de confusión (total 135 personas)

En relación a la información recopilada, se consiguió los siguientes ratios:

- Verdadero positivo (TP): 68/135 = **50.67%**
- Verdadero negativo (TN): 60/135 = **44.44%**
- Falso positivo (FP): 6/135 = **4.44%**
- Falso negativo (FT): 1/135 = **0.07%**

De allí, se determinó que tan confiable es la red neuronal mediante las siguientes fórmulas:

- Certeza: ¿Qué tan exacto es la red?
    - (TP/TN)/Total = 128/135 = **94.8%**
- Ratio de error: ¿Qué tanto se equivoca?
    - (FP+FN)/Total = 7/135 = **5.2%**
- Precisión:
    - (TP/Total Predicho Verdadero) = 68/69 = **98.55%**

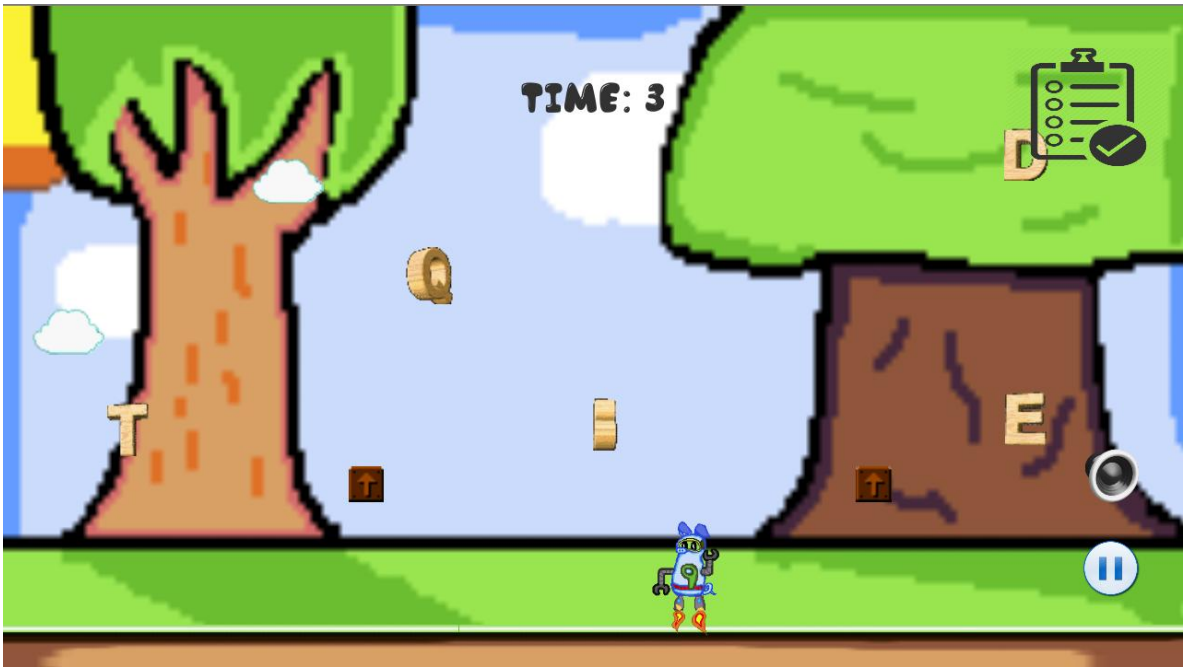
La red presenta un gran ratio de certeza, con solo un aproximado de 5% de error, y una precisión de una 98.55%. Aunque la red presenta algunos ciertos errores, la red muestra ser lo suficientemente confiable para la recopilación de la información relacionada acerca de la confirmación de las hipótesis, es decir, de lo que tan aceptado fue el videojuego como una forma de aprendizaje.

4.6 Resultados de la interfaz del videojuego:

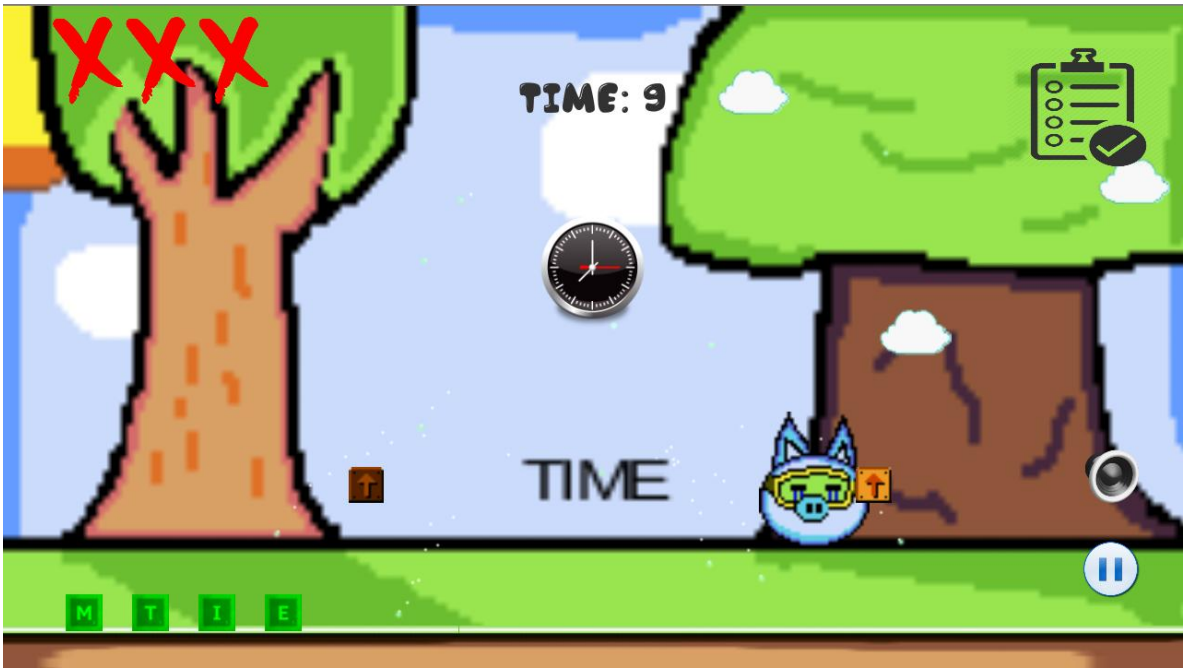
La interfaz del videojuego ha sido creada con el objetivo de apelar al público joven, por medio de la aplicación de un menú simple y fácil de comprender, con una gama de colores claros.

Cuando se inicia el nivel, se le mostrará al usuario una palabra al azar de la base de datos del mismo juego. Tendrá un límite de tiempo para memorizar la palabra,

aproximadamente diez segundos. Otros datos a memorizar es la traducción de la misma palabra y un archivo de sonido que reproduce la dicción de la palabra.



Una vez el tiempo expira, el jugador deberá recolectar las palabras en el orden correcto. Se contabiliza tanto el tiempo como que tan exacto es la palabra armada. Para subir su respuesta, el usuario deberá presionar la libreta para terminar el nivel.



Si el usuario cometió errores, estos se verán reflejados con Xs en la parte superior de la pantalla y con una imagen del cerdito triste y una imagen que representa a la palabra del nivel. Dependiendo de la gravedad de los errores, la red neuronal debe forzar al usuario a repetir la palabra o, si no lo es, pasará a examinar otra palabra de la base de datos.

El propósito de la interfaz y expresiones animadas del prototipo consiste en captar la atención del usuario con el motivo de mantener la atención del usuario y para motivarlo a continuar con el videojuego, a pesar de sus errores. Recompensas sutiles como lograr revelar la imagen feliz y el sonido de éxito del juego también alentar a mantener un ánimo constante para que el usuario siga practicando y aprendiendo nuevo vocabulario por medio del juego.

CAPÍTULO V: DISCUSIÓN, CONCLUSIONES Y RECOMENDACIONES

5.1 Discusión:

Según lo demostrado por los resultados encontrados por las encuestas, en el caso de la hipótesis H1, se ha determinado que la dificultad de los niveles está relacionada al mejor aprendizaje del usuario. En sí, esto se debe a que los usuarios están más propensos a involucrarse o sentir más concentrados en lo que se les presenta cuando este se presenta de manera de reto, los cuales es lo que se tratan los juegos en sí, una colección de retos (Granic, I. et al., 2014) presentados para el usuario con el propósito de entretener.

La hipótesis H2 se comprueba por medio de las respuestas brindadas en la encuesta en relación a la parte audiovisual del juego, es decir, de que tan efectivo logro ser su inclusión para la motivación del aprendizaje. De lo encontrado, se determina que la recepción de información medios auditivos o visuales resulta ser tan o más efectivo que simplemente leer la información. La captación de conocimiento se ha vuelto más audiovisual en este entorno moderno, con el uso de métodos de comunicación como el Internet, entre otros.

El caso de la hipótesis H3 esta cercanamente relacionado con el resultado de la hipótesis H2, el cual concierne a la cantidad de resultados acertados en la encuesta y el impacto de la enseñanza. En su mayoría (más del 90%), los resultados fueron correctos y las opiniones acerca de la efectividad de los videojuegos en la enseñanza, bajo ponderación, resulto ser mayor al 80% en todos los casos.

5.2 Conclusiones:

El objetivo del presente trabajo era demostrar la efectividad y facilidad de brindar enseñanza del vocabulario en inglés para una población objetivo de edad entre 5 a 11. Se tomó una muestra de 144 personas como la población a evaluar. La idea se centró en la enseñanza de idiomas, pero el concepto puede ir más allá de lo presentado, abarcando más temas de enseñanza como matemáticas y ciencias, pero de una manera más especializada, con el

propósito de enseñar y entretener al mismo tiempo. Cabe resaltar que la aplicación no solo es posible codificar para Android exclusivamente, sino también para sistemas operativos como iOS e incluso para computadoras en el hogar si existe una demanda alta por este tipo de enseñanza. A partir de esta idea, se desarrolló el prototipo y la investigación, en base al contexto peruano actual.

La aplicación de las redes neuronales para la predicción de los siguientes niveles, dan un nivel más profesional al tipo de trabajo realizado, además de ser confiable al tener un 95% de exactitud en sus resultados.

Se debe recalcar la efectividad de las respuestas de los usuarios. Los usuarios que hicieron la prueba de la encuesta se mostraron entretenidos durante las pruebas, y respondieron con claridad las respuestas, con una cantidad mínima de errores detectados, para nuestra sorpresa. En relación al tema del aspecto del tiempo utilizado para la resolución de los niveles, se hizo las preguntas correspondientes y se revisó las reseñas de los usuarios.

El problema residía en los controles del aplicativo, que resultaban ser complejos para el usuario en general. Sin embargo, como se dijo previamente, el ratio de respuestas correctas no se vio afectado en lo absoluto, aunque la dificultad de los controles interrumpía el sentido de inmersión en el usuario.

Finalmente, se planteó que el uso de técnicas de SCRUM agilizo en gran parte el desarrollo del proyecto, por medio de definición de hitos claves en un periodo casi semanal.

5.3 Recomendaciones:

A partir de lo concluido, se puede determinar con certeza que el uso de los videojuegos para transmitir enseñanza resulta ser más efectiva que los métodos convencionales. Sin embargo, como se demostró en las pruebas, los controles de la aplicación en cuestión tienen que tener controles adecuados para la población objetivo en cuestión. Si no se llega a cumplir esto, el desempeño, inmersión y ratio de aprendizaje del usuario se verá afectado.

Una recomendación en general para personas que quisieran adentrarse en este rubro poco explorado es determinar, previamente de realizar el aplicativo, es de fijar en claro a qué

público el aplicativo va a ir dirigido. Al tener en mente esto, el reto consiste en captar la atención e interés de dicho público. Las formas de captar la atención de personas entre el rango de edad de 5 a 11 años resultan ser considerablemente diferentes de las maneras de captar la atención de personas universitarias o profesionales.

Finalmente, se recomienda que sea posible aplicar mejores técnicas de ingeniería de sistemas para proyectos más ambiciosos que tengan el mismo propósito educativo. Lo único que uno debe tener en cuenta es saber cómo adaptar el concepto a enseñar de la manera más creativa posible y a la vez se fieles a los conceptos de ingeniería que van a basarse, como se ha hecho en este proyecto. El rubro es realmente nuevo, en especial aquí en el Perú. El apoyo a nuestro país sería inmenso si esta tendencia se logra expandir para las diferentes ramas educativas para que no solo niños escolares puedan ser beneficiados, sino también estudiantes adolescentes e incluso universitarios.

BIBLIOGRAFIA

Anki (software), (s.f.). En Wikipedia. Recuperado el 18 de febrero de 2017 de https://en.wikipedia.org/wiki/Anki_(software)

Akkerman, S., Admiraal, W., & Huizenga, J. (2009). *Storification in history education: a mobile game in and about medieval Amsterdam*. Computers and Education, 52(2), 449-459

Alaimo, Diego Martín (2013). *Proyectos ágiles con Scrum: flexibilidad, aprendizaje, innovación y colaboración en contextos complejos*. - 1a ed. - Ciudad Autónoma de Buenos Aires : Kleer.

Andean Air Mail & PERUVIAN TIMES (4 de diciembre, 2013). Peru Ranks Last in OECD Education Assessment. Recuperado el 23 de enero del 2017, from http://www.peruviantimes.com/04/peru-ranks-last-in-oecd-education-assessment/20855/

Anderson, A. A., Warburton, W. A. (2012). *The impact of violent video games: An overview*, chapter in W. Warburton & D. Braunstein (Eds.), *Growing Up Fast and Furious: Reviewing the Impacts of Violent and Sexualised Media on Children*, (pp. 56-84). Annandale, NSW, Australia: The Federation Press

Ashrafa, H., Ghanei, F., Salamie, M. (2014) *The Impact of Online Games on Learning English Vocabulary*. Recuperado el 20 de enero del 2017 de: http://www.sciencedirect.com/science/article/pii/S1877042814025099

Base de datos (s.f.). En Wikipedia. Recuperado el 17 de febrero de 2017 de https://es.wikipedia.org/wiki/Base_de_datos

Blázquez, A. (2010). *Metodología de la enseñanza del inglés como segunda lengua*. *Innovación y experiencias educativas*, 1-8.

British Council of Educational Intelligence (Mayo 2015) English in Peru. An examination of policy, perceptions and influencing factors. Retrieved January 25, 2017, from https://ei.britishcouncil.org/sites/default/files/latin-america-research/English%20in%20Peru.pdf

C#. (s.f.). En Wikipedia. Recuperado el 16 de febrero de 2017 de http://en.wikipedia.org/wiki/ C_Sharp_(programming_language)

Ferguson C.J, Olson C. (2013). *Video game violence among 'vulnerable' populations: the impact of violent games on delinquency and bullying among children with clinically elevated depression or attention deficit symptoms*, Journal of Youth and Adolescence. DOI 10.1007/s10964-013-9986-5

Glorot, X., Bengio. Y,(s.f.). *Understanding the difficulty of training deep feedforward neural networks*. Recuperado de http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf.

Gottman, J. M. (1986). *The world of coordinated play: Same- and cross-sex friendship in young children*. Cambridge, England: Cambridge, University Press.

Granic, I., Lobel, A., Engels, R., C., M., E. (2014). *The benefits of playing video games, in American Psychologist*, Vol. 69, No. 1., Enero 2014, (pp.66-78)

Green, C. S., & Bavelier, D. (2012*). Learning, attentional control, and action video games. Current Biology*, 22, 197–206. doi:10.1016/j.cub.2012.02.012

Griffths, M. (2002). *The educational benefits of videogames*. Recuperado el 1 de enero del 2017, de http://sheu.org.uk/sites/sheu.org.uk/files/imagepicker/1/eh203mg.pdf

Hamari, J., Shernoff, D.J., Rowe, E., Coller, B., Asbell-Clarke, J., Edwards, T. (2016) *Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning*. Recuperado de: http://www.sciencedirect.com/science/article/pii/S074756321530056X

Haykin, S. (2009). *Neural networks and Learning Machines.* 3ra edición. Pearson.

Hsu, C.-Y., Tsai, C.-C., & Wang, H.-Y. (2012). *Facilitating third graders' acquisition of scientific concepts through digital game-based learning: the effects of self-explanation principles*. Asia-Pacific Education Researcher, 21(1), 71-82.

Jiménez, R. M. (1994). *Estrategias mnemotécnicas para la enseñanza y aprendizaje del vocabulario en inglés*. CL&E, 79-88.

Louis Walker (2015). *The impact of using Memrise on student perceptions of leaning Latin vocabulary and on long-term memory of words.*

Millington, Ian (2006) *Artificial Intelligence for Games*, The Morgan Kaufmann Series in Interactive 3D Technology.

Moya, A. J. & Jimenez, M.J. (2004) *El proceso de interlengua en el aprendizaje del inglés como lengua extranjera en edades tempranas*. Recuperado de http://www.um.es/glosasdidacticas/doc-es/10moya.pdf

O'Neil, H. F., Wainess, R., & Baker, E. L. (2005) Classification of learning outcomes: Evidence from the computer games literature. The Curriculum Journal, 16, 455-474.

Reynoso, H. (2000) *Los métodos de enseñanza de lenguas y las teorías de aprendizaje*. Recuperado el 12 de enero de: http://encuentrojournal.org/textos/11.15.pdf

Richards, Jack . C., & Rodgers, Theodore. S. (2014). *Approaches and Methods in Language Teaching.* Cambridge: Cambridge University Press.

R.Lara-Cabrera, Cotta (2014) *An analysis of the structure and evolution of the scientific collaboration network of computer intelligence in games*, 395, 523–536

Russell, J., Norvig, P. (2004) *Inteligencia Artificial. Un Enfoque Moderno*. Recuperado de: http://stpk.cs.rtu.lv/sites/all/files/stpk/materiali/mi/artificial%20intelligence%20a%20modern%20approach.pdf

Salceanu, C. (2014) *The Influence of Computer Games on Children's Development. Exploratory Study on the Attitudes of Parents.* Recuperado el 01 de febrero del 2017 de: http://www.sciencedirect.com/science/article/pii/S1877042814050368

Sarkar, D. (1995) Methods to speed up error back-propagation learning algorithm. *ACM Computing Sources*. *27*(4), 527-528.

Sprenger, M. (2005). *How to Teach so students remember*. Recuperado el 22 de diciembre del 2016, de http://site.ebrary.com/lib/esan/reader.action?docID=10081763

SQLite (s.f.). En Wikipedia. Recuperado el 17 de febrero del 2017 de https://es.wikipedia.org/wiki/SQLite

TalkEnglish.com. *Top 2000 words Vocabulary words.* Recuperado el 18 de diciembre del 2017 de http://www.talkenglish.com/vocabulary/top-2000-vocabulary.aspx

Tüzün, H., Yilmaz-Soylu, M., Karakus¸ T., Inal, Y., & Kizilkaya, G. (2009*). The effects of computer games on primary school students' achievement and motivation in geography learning. Computers and Education*, 52(1), 68-77

User Experience Design. En Wikipedia. Recuperado el 18 de febrero del 2017 de https://en.wikipedia.org/wiki/User_experience_design

Valenzuela, M. (Agosto, 2015). *Regla de Aprendizaje de Retropropagación. Sistemas Conexionistas y Evolutivos* (IA–5005).

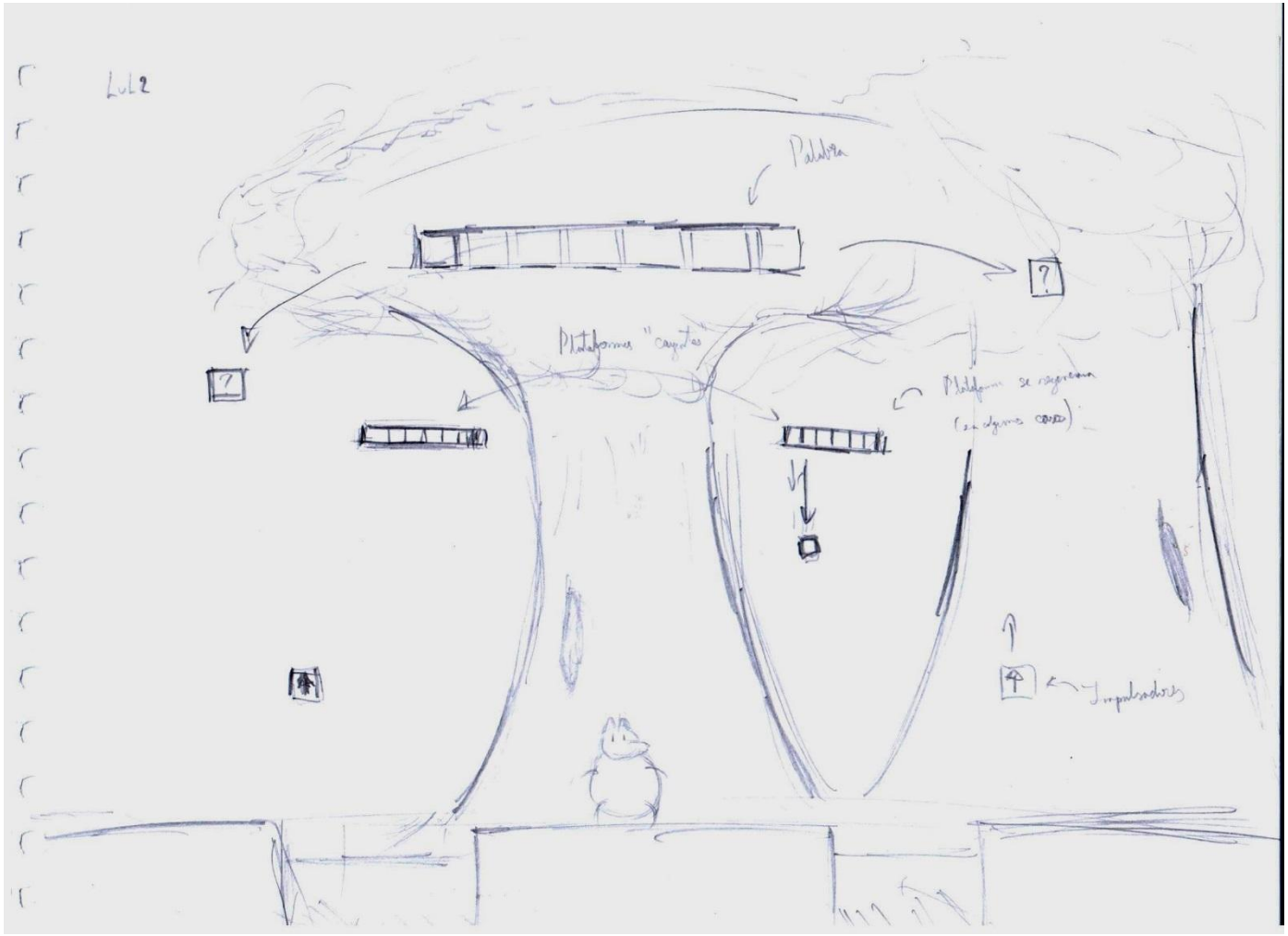
Vogel, J. J., Vogel, D. S., Cannon-Bowers, J., Bowers, C. A., Muse, K., & Wright, M. (2006). *Computer gaming and interactive simulations for learning: A meta-analysis. Journal of Educational Computing Research*, 34, 229 –243.

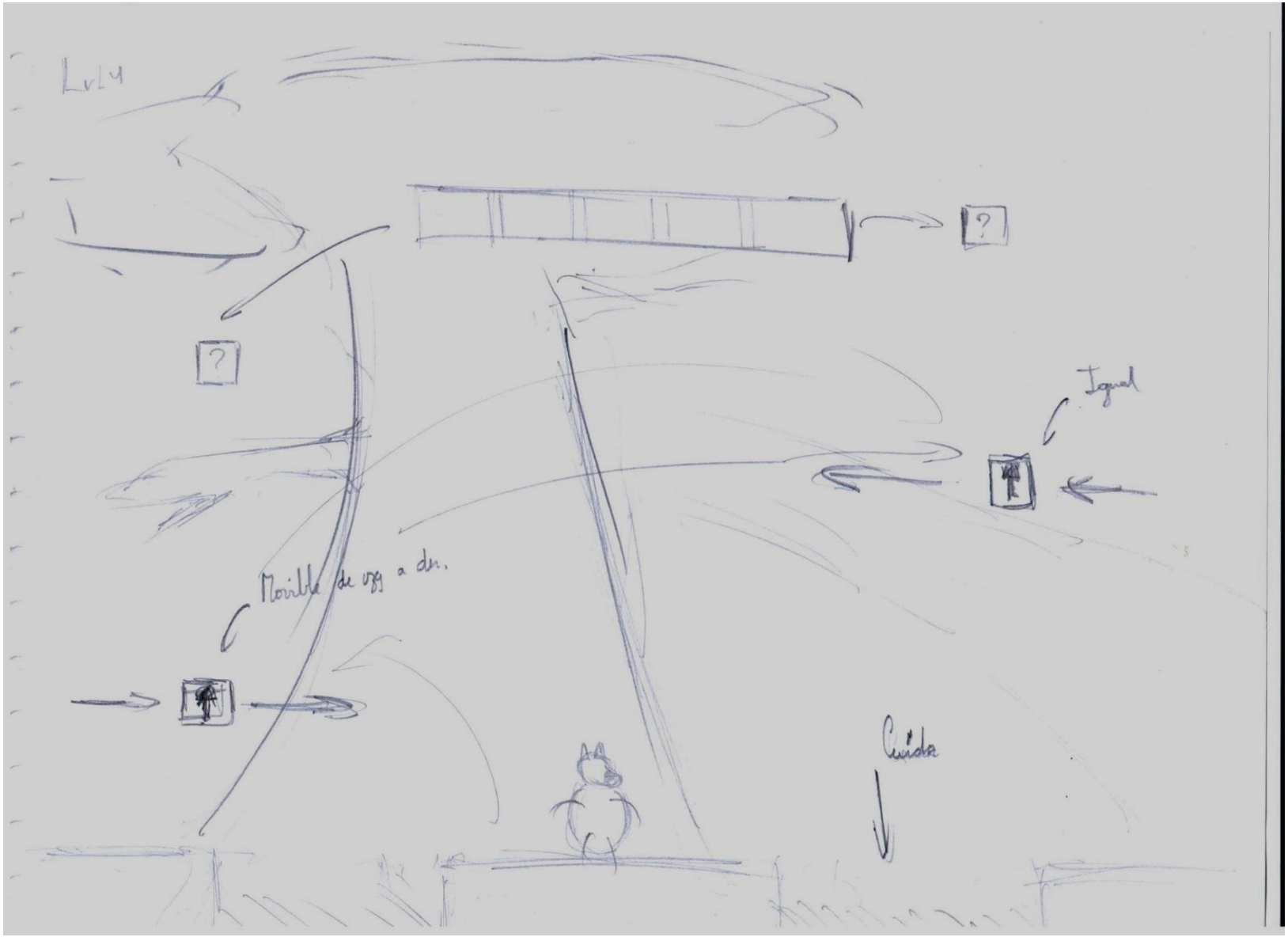
Yip, F.W.M., & Kwan, A.C.M.(2006). *Online vocabulary games as a tool for teaching and Learning English vocabulary*. Recuperado de: wiki.umd.edu/teamill/images/9/9f/Vicki's_article.

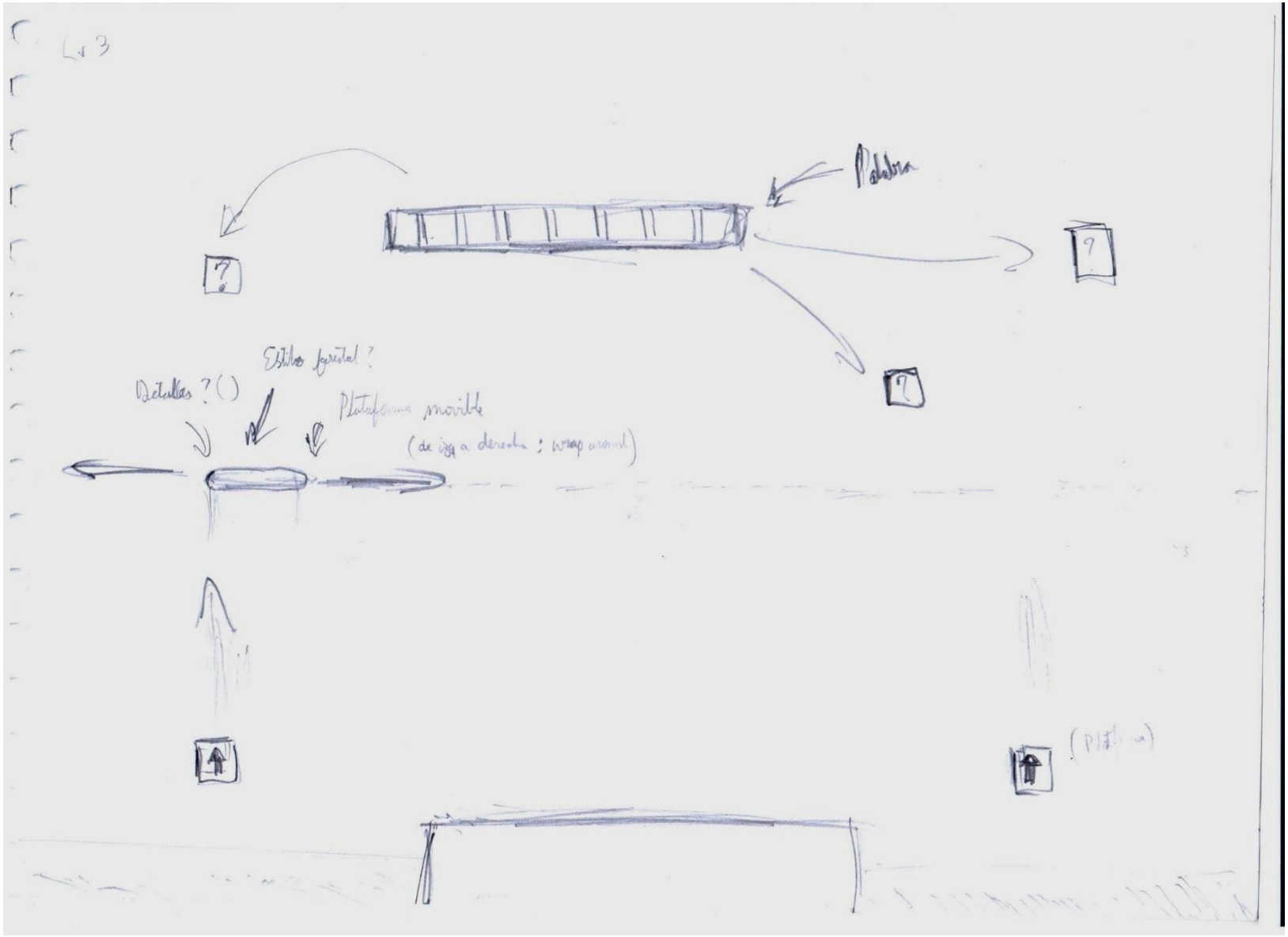
Yolageldili, G.,& Arikan, A. (2011). *Effectiveness of Using Games in Teaching Grammar to Young Learners. Elementary Education Online*, 10(1), 219-229, 2011. Recuperado de: http://ilkogretim-online.org.tr.

Yoke S., Wong & Hayati, M. (2014) *Computer Game As Learning and Teaching Tool For Object.* Recuperado el 23 de enero del 2017 de: http://ac.els-cdn.com/S1877042814014554/1-s2.0-S1877042814014554-main.pdf?_tid=3cd994a2-f179-11e6-ac20-00000aacb360&acdnat=1486941549_bd5847ed7edefcf34e97f256b5d4f013

ANEXOS


Lvl 2
Palabra
?
Plataformas "caen"
?
Plataformas se regeneran
(en algunos casos)
Impulsadores


Lvl 4
?
?
Igual
Movible de izq a der.
Caída


Lv 3
Palabra
?
?
?
Detalles ? ( )
Estilos tipográficos ?
Plataforma movible
(de izq a derecha : wrap around)

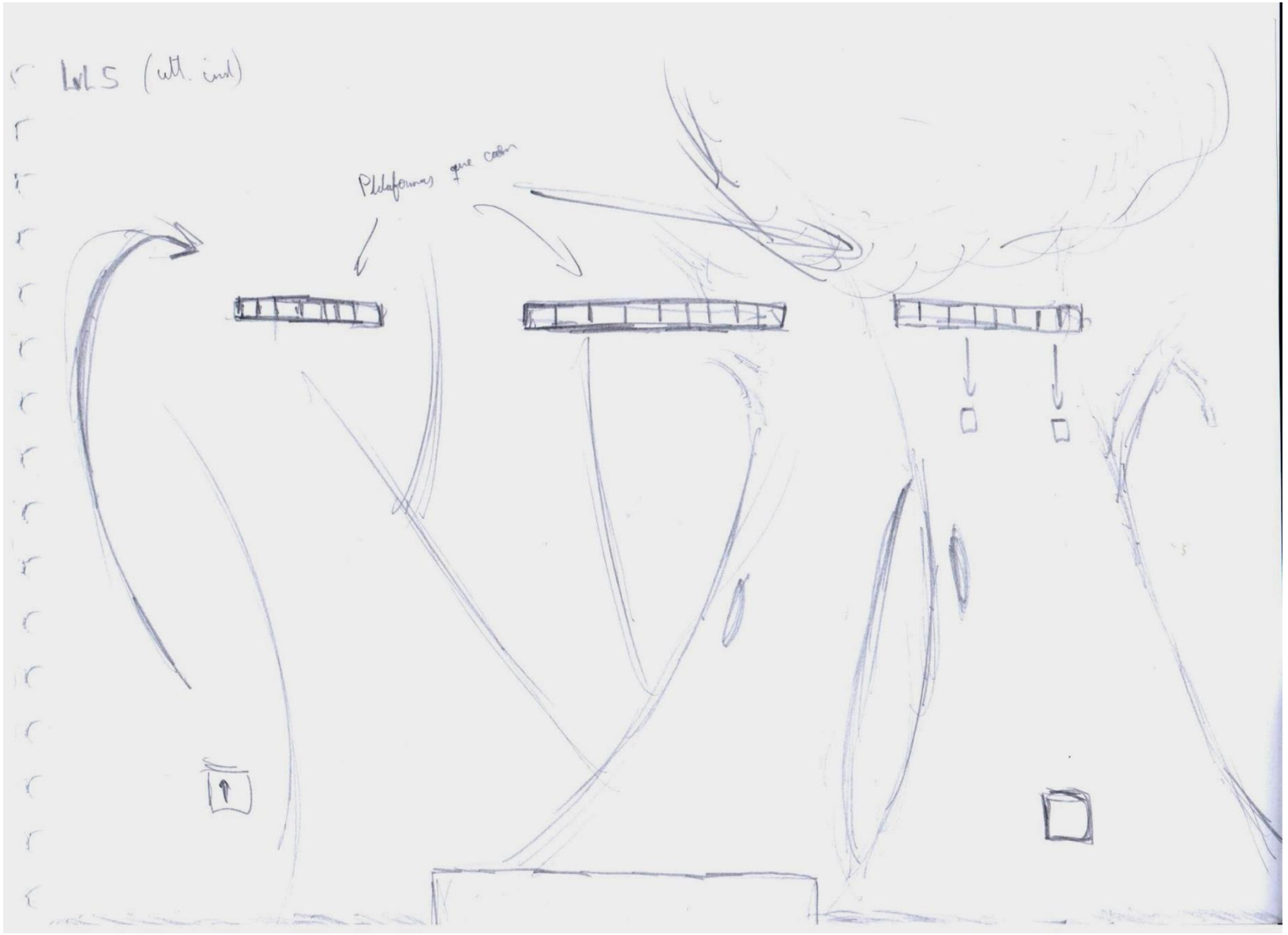

LvL 5 (ult. cust)
Plataformas que caen

Tabla de Entrenamiento:

La relación entre el resultado esperado y las palabras consiste en:

- Si el resultado esperado es 0 (es decir, no logro pasar la prueba del nivel como se esperaba), el algoritmo decide repetir la palabra y el nivel a la persona (Test)
- Si el resultado es 1, el algoritmo busca una nueva palabra de la base de datos para mostrarlo en la interfaz de juego para realizar un estudio de una nueva palabra. (Study)

Tiempo	Tipo de Exactitud	N° Inputs	Estado Previo	Resultado Esperado
11	1	4	1	1
13	3	3	1	1
14	1	3	0	1
26	3	14	1	0
18	2	1	0	1
4	1	5	0	0
27	3	9	0	0
25	3	2	0	0
30	2	12	1	0
3	2	7	1	1
28	3	15	0	0
23	1	15	1	0
7	1	6	0	0
22	3	8	1	0
11	1	3	0	1
20	2	12	0	1
18	2	11	0	1
1	2	15	0	1
4	1	10	0	0
28	2	5	1	1
3	2	5	1	1
29	2	2	1	1
6	1	8	1	1
1	3	11	0	0
28	3	2	1	0
29	3	9	1	0
22	3	12	0	0
1	2	15	1	1
14	2	3	1	1
20	1	1	1	1
24	3	4	0	0

4	2	11	0	1
20	2	2	1	1
26	2	14	1	0
5	3	7	0	0
29	2	4	0	1
5	1	10	0	0
3	3	4	0	0
6	3	7	0	0
16	1	4	0	1
25	3	12	0	0
25	3	13	0	0
17	3	11	0	0
14	2	15	1	1
26	1	8	0	0
3	1	12	1	1
15	1	7	0	1
7	2	9	0	1
1	1	4	0	0
27	1	13	0	0
14	3	4	1	1
2	3	5	0	0
9	2	5	0	1
22	1	15	1	0
13	3	1	0	0
7	2	10	1	1
25	2	15	0	0
10	1	1	0	0
22	1	9	1	0
3	1	15	1	1
13	2	4	0	1
2	3	9	1	1
28	1	10	0	0
9	1	6	0	0
22	3	12	1	0
13	3	11	1	0
5	1	10	1	1
22	3	2	0	0
17	2	5	0	1
14	2	2	0	1
30	2	5	0	1
21	1	5	1	0
3	2	1	1	1
6	3	3	0	0
21	2	13	0	0
19	2	11	0	1
18	2	15	1	1

4	2	1	0	1
17	2	7	0	1
30	1	7	0	0
2	3	4	0	0
9	1	4	0	0
1	3	8	0	0
10	2	11	0	1
23	1	7	1	0
1	2	6	0	1
8	3	3	0	0
30	3	14	1	0
3	3	15	1	0
29	1	11	1	0
1	2	3	1	1
2	2	14	1	1
12	1	5	0	1
20	1	6	0	1
8	2	8	1	1
19	1	6	1	1
9	2	4	1	1
26	1	5	1	0
4	2	7	0	1
16	2	13	0	1
27	3	8	1	0
25	1	13	0	0
16	1	9	0	1
7	2	4	1	1
14	3	8	1	1
12	3	6	0	0
24	1	10	1	0
21	3	11	1	0
7	1	4	1	1
5	3	15	0	0
6	2	6	1	1
30	3	15	1	0
2	2	3	0	1
19	3	1	1	1
18	3	4	1	1
1	1	11	1	1
21	3	8	1	0
17	3	12	1	0
8	3	9	1	1
17	3	10	1	1
25	2	10	0	1
4	1	4	1	1
28	2	11	1	0

19	2	10	0	1
16	2	5	0	1
8	2	1	1	1
28	2	8	1	1
10	1	3	1	1
6	2	7	1	1
17	1	15	1	0
2	2	6	0	1
13	3	3	0	0
13	1	6	1	1
15	1	12	0	0
18	3	10	0	0
4	2	7	1	1
30	2	13	1	0
8	3	4	1	1
1	3	5	0	0
29	1	4	0	0
13	2	3	0	1
25	2	6	0	1
23	1	12	1	0
18	2	10	1	1
30	3	6	0	0
12	3	12	0	0
27	2	10	0	1
11	3	11	1	0
14	2	14	0	1
3	1	1	1	1
14	2	9	0	1
15	2	5	1	1
2	2	5	1	1
23	3	2	0	0
30	3	7	0	0
28	3	7	1	0
6	1	14	1	1
7	3	14	0	0
26	3	8	0	0
21	2	2	0	1
17	2	10	1	1
10	3	11	0	0
29	3	8	0	0
3	3	13	1	0
27	3	7	1	0
5	2	8	1	1
27	3	3	1	0
10	2	13	1	1
21	1	7	0	0

20	3	13	1	0
4	3	4	1	1
7	1	12	1	1
27	2	15	0	0
12	2	10	1	1
1	1	14	0	0
6	2	15	1	1
20	2	11	1	1
20	1	2	1	1
7	1	2	1	1
15	1	14	1	0
6	2	10	0	1
9	2	13	0	1
18	2	14	1	1
21	1	4	1	0
3	2	4	1	1
13	1	7	0	1
5	2	12	1	1
24	2	13	0	0
18	1	14	0	0
5	1	4	0	0
12	3	5	0	0
22	2	15	1	0
8	3	7	1	1
3	3	9	0	0
30	3	9	0	0
6	3	3	1	1
20	1	7	1	1
21	1	15	1	0
19	2	8	0	1
21	1	7	1	0
6	2	11	0	1
4	2	4	1	1
7	3	15	0	0
24	3	5	1	0
8	1	14	1	1
7	1	11	1	1
8	3	8	1	1
15	3	4	0	0
13	1	13	0	0
13	2	5	1	1
14	3	12	1	0
26	3	1	1	0
19	3	14	1	0
29	2	9	0	1
13	3	10	0	0

9	3	4	0	0
5	1	2	0	0
15	2	14	0	1
20	1	6	1	1
10	3	8	0	0
24	3	14	1	0
8	1	1	0	0
7	3	7	1	1
6	3	2	0	0
25	1	5	1	0
12	3	3	0	0
3	2	6	0	1
10	2	3	1	1
21	3	6	1	0
26	2	14	0	0
24	3	14	0	0
19	1	13	0	0
9	2	1	0	1
29	2	7	0	1
27	2	13	0	0
5	1	14	0	0
9	2	2	1	1
12	2	5	1	1
10	3	14	1	0
23	1	10	0	0
26	3	11	1	0
8	1	1	1	1
11	1	4	0	1
12	1	7	0	1
2	2	7	1	1
26	3	13	1	0
12	1	10	0	1
24	3	15	1	0
8	3	2	1	1
28	1	7	1	0
29	2	15	1	0
12	2	1	0	1
2	1	4	0	0
30	2	2	0	1
17	3	4	1	1
19	2	10	1	1
5	2	7	1	1
22	3	2	1	0
20	3	2	1	1
7	2	4	0	1
11	2	6	0	1

3	1	2	0	0
18	2	13	1	1
11	3	8	1	1
4	3	6	1	1
9	3	15	0	0
25	2	10	1	1
3	2	4	0	1
10	2	6	1	1
7	2	2	1	1
26	1	11	0	0
21	3	2	0	0
11	1	13	0	0
22	2	7	1	1
19	3	3	1	1
13	1	9	0	1
4	3	12	1	0
6	3	10	0	0
5	1	12	1	1
7	1	14	0	0
16	2	13	1	1
15	1	6	1	1
22	2	9	0	1
17	2	1	0	1
15	1	3	1	1
11	2	11	0	1
7	3	4	0	0
15	2	6	0	1
4	3	1	0	0
7	3	5	1	1
5	1	7	1	1
5	1	1	1	1
26	1	10	0	0
22	1	7	0	0
10	1	8	1	1
25	2	3	0	1
3	3	2	1	1
25	1	3	0	0
30	2	1	1	1
20	2	4	0	1
15	1	7	1	1
30	2	7	0	1
24	3	7	1	0
16	2	8	1	1
17	2	15	1	1
23	1	14	1	0
13	1	12	1	0

1	2	9	1	1
12	1	2	0	1
1	1	5	1	1
19	1	14	1	0
18	1	2	1	1
29	3	1	1	0
23	2	8	0	1
29	1	11	0	0
4	1	11	1	1
13	1	15	1	0
3	3	6	0	0
28	2	12	1	0
30	2	12	0	0
11	1	15	0	0
9	1	2	0	0
30	2	14	0	0
6	3	8	0	0
7	2	5	0	1
28	1	14	1	0
11	1	15	1	0
26	3	8	1	0
3	3	14	0	0
2	3	5	1	1
9	1	5	1	1
14	1	15	1	0
8	2	15	1	1
14	2	6	1	1
10	1	2	0	0
2	2	10	1	1
11	1	6	1	1
19	2	1	0	1
21	1	14	1	0
11	3	7	1	1
25	2	6	1	1
5	3	11	1	0
16	3	8	1	1
19	1	11	0	0
19	1	2	1	1
19	1	5	0	1
24	2	2	1	1
27	1	10	1	0
8	2	15	0	1
14	3	12	0	0
4	3	13	0	0
6	3	9	1	1
27	2	4	1	1

18	1	15	1	0
27	3	14	0	0
23	1	8	1	0
11	2	7	0	1
11	2	5	0	1
4	2	12	1	1
27	2	8	0	1
3	2	2	1	1
23	2	3	1	1
16	2	11	0	1
1	3	8	1	1
23	3	6	1	0
16	1	8	1	1
18	2	2	0	1
25	3	3	1	0
17	1	5	0	1
1	1	3	1	1
29	2	5	1	1
12	3	13	0	0
3	1	11	1	1
29	3	11	1	0
22	2	5	1	1
23	3	4	0	0
19	2	13	1	1
9	1	6	1	1
2	2	13	1	1
1	1	2	0	0
30	1	6	0	0
26	2	13	0	0
21	2	12	1	0
27	1	8	1	0
12	2	4	1	1
13	2	1	0	1
10	3	4	0	0
23	1	3	0	0
9	3	9	1	1
6	1	6	1	1
17	1	8	1	1
3	1	3	0	0
25	3	13	1	0
26	1	4	1	0
21	3	3	1	0
8	3	8	0	0
22	3	7	0	0
13	3	12	1	0
27	1	1	1	0

24	2	5	0	1
20	2	1	0	1
19	3	7	0	0
28	2	7	1	1
22	1	11	0	0
1	2	13	1	1
11	3	5	0	0
23	3	12	0	0
27	3	13	0	0
25	1	5	0	0
3	3	15	0	0
6	2	3	1	1
18	3	1	1	1
23	3	15	0	0
21	2	6	0	1
24	3	6	0	0
13	2	7	1	1
24	1	8	1	0
28	2	3	0	1
29	3	10	1	0
17	1	11	0	0
15	1	6	0	1
5	2	3	1	1
3	3	3	1	1
1	2	10	0	1
5	1	7	0	0
25	2	11	0	0
19	1	1	0	1
9	2	2	0	1
28	1	12	1	0
5	1	3	0	0
3	3	1	0	0
10	3	12	1	0
19	3	11	0	0
10	3	9	1	1
3	3	13	0	0
27	3	5	1	0
14	3	5	1	1
10	1	10	1	1
23	2	1	1	1
18	3	7	0	0
12	3	9	0	0
26	3	10	0	0
13	3	13	0	0
30	3	15	0	0
26	3	2	1	0

23	3	14	0	0
30	3	13	1	0
26	1	7	1	0
8	2	6	0	1
17	1	12	1	0
5	3	5	1	1
16	1	4	1	1
12	1	15	1	0
20	3	9	0	0
28	3	4	1	0
28	2	1	1	1
18	1	13	0	0
7	2	5	1	1
1	3	3	1	1
25	1	4	0	0
7	1	5	1	1
4	2	14	0	1
24	2	7	1	1
29	1	7	1	0
6	1	7	0	0
27	2	12	1	0
29	3	2	0	0
1	3	10	1	1
16	3	8	0	0
28	1	13	1	0
22	3	3	0	0
27	1	1	0	0
30	2	8	1	1
18	1	3	1	1
23	1	13	0	0
11	3	6	1	1
22	1	1	0	0
18	2	5	1	1
24	3	9	1	0
8	3	15	1	0
30	2	3	0	1
9	1	5	0	0
26	2	1	0	1
16	1	12	1	0
21	2	13	1	0
28	3	6	0	0
20	2	13	0	1
1	2	9	0	1
30	2	10	0	1
9	3	5	0	0
13	1	15	0	0

9	1	10	0	0
12	1	11	0	0
8	2	1	0	1
8	2	4	1	1
10	2	1	1	1
25	1	10	1	0
16	3	6	0	0
5	3	6	1	1
29	3	8	1	0
1	2	10	1	1
5	2	11	0	1
1	3	2	1	1
30	3	4	1	0
18	2	14	0	1
5	2	14	0	1
19	3	5	0	0
2	1	10	0	0
3	1	9	1	1
12	1	6	0	1
18	1	7	1	1
18	2	6	0	1
14	1	7	0	1
1	2	14	1	1
2	2	15	0	1
24	2	11	0	0
8	2	4	0	1
3	1	1	0	0
29	1	7	0	0
9	2	15	1	1
13	2	7	0	1
5	1	9	1	1
10	3	7	1	1
10	1	9	0	0
25	1	11	0	0
7	2	1	0	1
15	3	3	1	1
18	2	3	0	1
17	2	8	1	1
14	3	14	1	0
9	1	12	0	0
26	3	6	0	0
26	3	12	1	0
15	2	15	0	1
15	2	9	1	1
16	1	12	0	0
17	2	4	0	1

15	3	2	0	0
1	1	11	0	0
14	1	4	0	1
30	1	9	0	0
7	2	12	0	1
12	2	12	0	1
6	2	9	0	1
6	2	5	0	1
27	3	8	0	0
20	3	13	0	0
3	1	15	0	0
3	3	12	1	0
14	2	11	1	1
19	2	15	0	1
14	3	6	1	1
7	2	7	1	1
30	2	11	1	0
5	1	8	0	0
4	1	3	0	0
20	3	3	1	1
14	1	10	1	1
1	3	4	0	0
3	2	1	0	1
29	2	8	1	1
24	1	7	1	0
5	3	5	0	0
16	1	11	1	0
20	1	8	1	1
9	2	7	1	1
18	3	3	0	0
20	3	4	0	0
11	3	9	1	1
2	1	8	0	0
30	3	3	0	0
25	2	8	1	1
14	1	12	1	0
24	3	12	0	0
12	2	7	0	1
8	2	13	0	1
7	3	9	1	1
22	2	2	0	1
21	1	8	1	0
3	2	15	1	1
6	2	9	1	1
15	3	14	0	0
20	2	6	0	1

25	1	13	1	0
13	2	15	1	1
8	3	13	0	0
1	1	13	1	1
1	2	12	0	1
1	3	6	0	0
29	3	12	1	0
17	3	4	0	0
9	1	7	1	1
8	3	9	0	0
6	2	4	0	1
3	3	10	1	1
14	1	11	1	0
9	3	10	0	0
21	2	15	0	0
22	2	7	0	1
8	3	11	0	0
10	3	3	0	0
1	1	8	0	0
19	2	14	0	1
20	3	1	1	1
18	1	6	1	1
1	3	13	0	0
5	2	10	0	1
27	1	14	0	0
19	1	15	0	0
25	1	9	0	0
17	1	4	1	1
14	3	15	1	0
23	3	1	1	0
30	1	14	1	0
20	3	7	1	1
28	1	6	1	0
20	2	1	1	1
20	3	14	0	0
12	2	6	1	1
11	3	2	0	0
10	3	5	1	1
3	1	10	0	0
8	3	14	1	0
5	3	13	0	0
3	3	11	0	0
7	1	12	0	0
13	3	14	0	0
28	1	2	0	0
26	1	5	0	0

19	2	4	0	1
2	1	1	1	1
8	1	3	0	0
2	1	1	0	0
26	1	14	1	0
25	1	7	1	0
15	2	4	0	1
26	2	11	0	0
27	2	3	0	1
6	3	13	1	0
30	1	11	1	0
29	2	11	0	0
29	3	15	0	0
24	2	13	1	0
2	1	2	0	0
17	1	3	1	1
5	1	6	1	1
22	1	6	1	0
22	2	4	0	1
30	2	13	0	0
21	3	3	0	0
1	1	10	1	1
2	2	11	0	1
16	3	9	1	1
8	3	4	0	0
13	2	2	0	1
15	1	10	0	1
7	2	2	0	1
8	1	4	1	1
16	2	2	0	1
2	1	7	0	0
7	2	3	0	1
8	2	14	1	1
25	1	12	0	0
29	3	14	1	0
17	3	7	1	1
21	2	1	1	1
29	3	5	0	0
23	1	2	1	0
30	2	6	0	1
30	1	2	0	0
30	3	2	0	0
6	3	2	1	1
27	1	6	1	0
27	3	12	0	0
29	2	5	0	1

23	2	2	1	1
7	3	5	0	0
27	1	11	0	0
10	2	6	0	1
21	2	9	0	1
9	1	11	0	0
24	3	1	1	0
28	1	11	0	0
10	2	2	1	1
13	1	13	1	0
15	3	8	0	0
4	3	7	0	0
18	3	5	0	0
2	2	3	1	1
17	2	9	1	1
17	3	5	1	1
16	2	14	0	1
17	1	2	1	1
11	3	8	0	0
14	3	9	0	0
10	2	5	1	1
5	3	9	0	0
14	3	1	1	1
14	1	5	1	1
25	2	13	1	0
24	1	11	0	0
27	1	14	1	0
30	3	10	1	0
25	2	14	0	0
18	1	1	1	1
23	1	6	1	0
29	1	1	0	0
2	1	13	0	0
3	1	9	0	0
8	3	13	1	0
11	2	10	1	1
9	2	15	0	1
14	2	7	0	1
18	2	1	1	1
29	3	1	0	0
24	1	9	1	0
4	3	4	0	0
3	1	12	0	0
19	3	10	1	1
5	1	5	0	0
20	3	11	1	0

18	2	4	1	1
4	2	3	0	1
21	2	1	0	1
7	2	10	0	1
25	2	9	1	1
21	2	10	1	1
2	3	10	1	1
28	2	2	1	1
21	3	4	0	0
9	3	1	1	1
4	1	12	0	0
11	2	4	0	1
17	2	10	0	1
30	3	9	1	0
3	1	11	0	0
25	3	7	0	0
17	3	1	0	0
1	1	7	0	0
15	2	10	1	1
18	3	5	1	1
15	2	15	1	1
25	2	3	1	1
29	1	13	1	0
27	1	5	1	0
22	1	8	0	0
20	3	6	0	0
17	3	2	1	1
16	2	8	0	1
14	3	3	1	1
29	1	8	1	0
20	3	5	0	0
21	1	6	1	0
1	2	2	1	1
1	3	12	0	0
22	3	4	0	0
21	1	12	1	0
23	3	5	0	0
8	1	5	1	1
15	3	12	0	0
13	1	11	1	0
4	2	8	1	1
14	2	8	1	1
16	1	7	1	1
26	3	5	0	0
5	3	6	0	0
14	3	7	0	0

17	1	9	1	1
10	1	12	1	1
11	1	11	1	0
28	2	15	0	0
24	2	8	1	1
30	2	1	0	1
13	3	6	1	1
27	1	4	1	0
23	3	7	1	0
7	1	14	1	1
12	3	1	0	0
25	1	1	1	0
21	3	2	1	0
1	1	15	1	1
11	3	3	1	1
28	1	4	1	0
4	1	10	1	1
22	2	10	0	1
21	1	4	0	0
18	2	10	0	1
23	1	5	0	0
27	2	7	1	1
16	1	8	0	1
21	3	12	0	0
21	3	11	0	0
20	3	10	0	0
23	2	14	0	0
7	1	9	0	0
22	2	13	0	0
27	3	5	0	0
13	2	9	0	1
24	3	1	0	0
10	1	3	0	0
8	2	14	0	1
3	3	12	0	0
14	1	7	1	1
19	2	15	1	1
1	1	6	0	0
6	3	10	1	1
7	1	7	0	0
6	3	12	0	0
12	1	13	1	0
28	1	7	0	0
27	1	12	0	0
17	3	6	0	0
2	1	6	0	0

26	3	11	0	0
28	1	9	1	0
7	3	13	1	0
10	3	6	0	0
21	2	8	1	1
29	1	9	0	0
18	3	6	0	0
3	3	2	0	0
20	3	8	1	1
9	3	5	1	1
5	1	9	0	0
1	1	6	1	1
27	2	2	1	1
7	2	13	1	1
27	2	10	1	1
14	3	1	0	0
20	3	12	1	0
8	3	7	0	0
3	1	13	0	0
6	1	7	1	1
3	1	5	1	1
12	2	3	1	1
29	2	4	1	1
11	1	3	1	1
18	3	2	0	0
26	2	8	0	1
24	2	4	1	1
11	1	7	0	1
8	1	10	0	0
22	1	14	1	0
3	2	3	1	1
26	2	4	1	1
23	1	10	1	0
23	2	9	1	1
20	3	7	0	0
13	2	2	1	1
29	1	8	0	0
28	3	8	1	0
28	3	13	0	0
17	1	6	1	1
10	1	14	1	1
16	2	3	1	1
19	3	9	0	0
12	3	4	1	1
10	3	3	1	1
3	1	13	1	1

14	1	5	0	1
7	1	10	0	0
5	2	10	1	1
26	1	8	1	0
10	1	1	1	1
1	3	15	0	0
28	3	1	1	0
25	3	10	1	0
29	2	6	1	1
21	3	5	1	0
22	2	8	1	1
10	2	3	0	1
26	1	1	0	0
10	1	14	0	0
30	2	14	1	0
21	3	14	0	0
19	1	10	0	1
12	3	7	1	1
14	2	3	0	1
13	1	11	0	0
28	2	9	0	1
14	3	10	1	1
9	1	1	1	1
6	2	1	0	1
10	3	14	0	0
27	2	6	0	1
4	3	5	1	1
8	2	6	1	1
17	2	6	1	1
26	2	12	0	0
19	2	3	1	1
28	2	4	0	1
25	3	4	0	0
1	2	11	0	1
4	3	10	1	1
22	3	11	0	0
13	2	12	0	1
12	1	3	0	1
3	2	11	1	1
17	2	2	0	1
22	2	3	1	1
4	1	12	1	1
5	2	9	1	1
14	3	11	0	0
20	3	5	1	1
7	2	7	0	1

21	1	3	0	0
5	1	5	1	1
13	3	11	0	0
10	2	10	0	1
1	3	12	1	0
21	1	9	0	0
14	2	12	1	1
6	2	14	1	1
8	3	10	0	0
22	2	1	0	1
6	3	15	1	0
5	2	11	1	1
16	2	14	1	1
10	3	12	0	0
21	2	15	1	0
14	3	6	0	0
24	2	10	0	1
21	3	1	0	0
3	2	5	0	1
16	3	11	1	0
7	3	6	1	1
20	2	14	1	1
23	2	2	0	1
5	2	15	1	1
12	1	1	1	1
2	3	13	0	0
24	3	8	1	0
11	2	2	1	1
30	3	11	0	0
16	3	13	0	0
24	1	7	0	0
15	1	2	1	1
5	2	6	1	1
12	2	9	1	1
26	2	10	1	1
13	1	4	0	1
9	2	8	1	1
11	2	1	1	1
17	2	11	1	1
11	3	12	0	0
24	1	12	1	0
12	1	9	1	1
19	1	4	1	1
17	1	14	0	0
11	3	2	1	1
3	1	14	0	0

30	2	2	1	1
4	3	6	0	0
18	1	8	1	1
2	3	8	0	0
11	1	13	1	0
13	3	10	1	1
7	2	14	1	1
11	3	4	0	0
15	1	15	0	0
15	1	11	1	0
29	2	15	0	0
11	2	15	0	1
16	3	7	1	1
9	2	3	0	1
18	1	5	0	1
13	3	8	0	0
28	1	6	0	0
28	2	12	0	0
8	1	7	0	0
28	3	3	1	0
23	2	4	0	1
9	1	10	1	1
11	1	9	1	1
25	3	9	1	0
26	2	7	1	1
28	2	10	0	1
10	3	10	0	0
14	1	9	1	1
10	3	8	1	1
17	2	14	1	1
13	1	2	1	1
28	1	10	1	0
16	1	3	0	1
21	1	2	1	0
23	1	7	0	0
16	3	14	0	0
8	3	2	0	0
17	2	15	0	1
6	1	11	1	1
27	3	10	0	0
7	3	3	1	1
20	1	3	0	1
4	1	14	1	1
27	1	13	1	0
12	2	8	0	1
1	1	9	1	1

15	2	9	0	1
22	3	9	0	0
8	2	13	1	1
5	2	14	1	1
21	3	10	1	0
27	2	9	0	1
1	2	12	1	1
14	2	10	0	1
17	2	8	0	1
10	2	11	1	1
3	1	10	1	1
30	2	4	1	1
12	2	7	1	1
1	2	4	0	1
29	1	13	0	0
30	3	5	1	0
6	2	5	1	1
22	1	14	0	0
26	1	6	0	0
16	1	6	1	1
24	1	9	0	0
2	3	6	1	1
11	2	5	1	1
26	1	1	1	0
1	2	7	0	1
9	1	11	1	1
30	3	3	1	0
17	1	9	0	1
11	2	15	1	1
8	2	3	0	1
15	3	8	1	1
7	1	8	1	1
29	1	2	0	0
30	1	12	0	0
17	2	4	1	1
3	2	14	1	1
19	2	5	0	1
8	3	11	1	0
14	1	6	0	1
21	3	13	0	0
24	1	6	0	0
7	3	8	0	0
30	3	6	1	0
2	1	12	1	1
16	1	5	1	1
21	1	1	1	0

15	2	6	1	1
15	2	3	0	1
25	3	11	0	0
11	1	5	0	1
23	3	6	0	0
5	3	4	1	1
15	1	10	1	1
25	1	8	1	0
6	2	12	0	1
2	1	11	1	1
29	1	4	1	0
27	3	4	1	0
30	1	4	1	0
26	3	15	1	0
15	3	5	1	1
5	1	6	0	0
28	2	1	0	1
22	3	11	1	0
7	3	12	0	0
2	2	7	0	1
2	1	12	0	0
15	3	4	1	1
3	2	7	0	1
17	2	12	1	1
5	2	1	1	1
3	2	3	0	1
15	3	13	0	0
6	3	13	0	0
6	3	1	0	0
17	3	10	0	0
4	3	8	1	1
3	3	3	0	0
4	3	14	1	0
24	1	14	1	0
1	3	10	0	0
23	1	1	0	0
4	1	7	0	0
24	3	13	0	0
26	2	12	1	0
7	1	2	0	0
7	1	11	0	0
28	1	15	0	0
18	3	12	1	0
12	1	6	1	1
6	1	4	1	1
6	2	11	1	1

22	2	14	0	0
28	3	7	0	0
28	3	4	0	0
16	3	13	1	0
26	3	7	1	0
27	2	1	1	1
11	3	6	0	0
18	2	5	0	1
6	3	14	0	0
24	3	11	1	0
6	3	11	0	0
2	3	13	1	0
18	1	9	1	1
24	3	6	1	0
30	3	13	0	0
9	3	6	1	1
10	2	15	0	1
21	3	7	0	0
28	1	8	0	0
15	1	5	1	1
29	2	1	0	1
25	1	1	0	0
25	3	3	0	0
20	1	5	1	1
9	1	13	0	0
27	3	3	0	0
22	3	7	1	0
7	3	4	1	1
24	3	11	0	0
13	1	9	1	1
16	2	15	0	1
10	3	4	1	1
7	3	1	1	1
25	1	8	0	0
25	1	9	1	0
7	2	8	1	1
20	3	9	1	1
27	3	10	1	0
17	1	1	1	1
24	3	4	1	0
4	2	6	0	1
22	2	2	1	1
23	3	2	1	0
14	3	11	1	0
9	2	7	0	1
5	1	13	0	0

8	3	5	1	1
13	1	10	0	1
24	2	15	0	0
29	3	15	1	0
12	2	14	0	1
25	2	12	1	0
22	2	6	1	1
5	3	3	0	0
15	2	12	1	1
16	2	11	1	1
16	3	9	0	0
23	2	5	1	1
6	3	4	1	1
28	1	3	0	0
19	1	3	0	1
26	2	3	0	1
12	3	13	1	0
13	1	6	0	1
12	2	9	0	1
4	1	3	1	1
16	3	4	0	0
2	3	14	1	0
11	2	11	1	1
25	3	1	0	0
1	3	5	1	1
12	1	5	1	1
25	2	13	0	0
30	1	13	0	0
11	1	5	1	1
25	3	5	0	0
3	3	9	1	1
21	2	5	1	1
24	1	6	1	0
16	3	3	0	0
24	2	8	0	1
15	1	8	0	1
8	3	1	1	1
17	1	5	1	1
30	1	13	1	0
10	2	13	0	1
5	1	11	1	1
6	1	11	0	0
24	2	9	0	1
20	2	11	0	1
14	2	11	0	1
16	2	3	0	1

10	2	7	1	1
12	3	1	1	1
15	2	5	0	1
15	3	10	0	0
7	3	9	0	0
1	3	1	1	1
29	3	14	0	0
14	2	1	1	1
18	3	14	0	0
15	3	2	1	1
4	3	7	1	1
20	2	10	1	1
22	2	13	1	0
19	1	6	0	1
11	3	10	1	1
19	1	10	1	1
4	1	8	0	0
19	1	1	1	1
14	2	14	1	1
12	3	10	0	0
14	2	13	1	1
4	1	8	1	1
12	1	2	1	1
26	1	13	1	0
30	1	15	0	0
26	2	9	0	1
15	3	1	1	1
1	2	2	0	1
15	3	10	1	1
4	1	13	0	0
30	1	3	1	0
14	1	1	1	1
16	1	6	0	1
9	1	12	1	1
22	3	6	0	0
15	3	11	1	0
16	3	15	1	0
24	2	14	0	0
15	1	4	0	1
2	2	5	0	1
27	2	3	1	1
28	1	8	1	0
10	1	11	0	0
19	3	15	1	0
2	2	1	0	1
7	1	3	1	1

24	1	3	0	0
1	2	7	1	1
15	2	1	1	1
22	1	1	1	0
12	2	1	1	1
24	1	8	0	0
16	2	6	0	1
24	2	9	1	1
11	3	14	1	0
21	3	7	1	0
10	1	6	0	0
13	3	6	0	0
23	1	9	0	0
26	1	4	0	0
22	1	13	0	0
14	2	2	1	1
12	3	2	0	0
5	1	13	1	1
28	3	9	0	0
23	2	9	0	1
27	3	7	0	0
5	2	13	0	1
8	2	12	1	1
11	1	2	1	1
23	2	6	0	1
21	1	8	0	0
18	1	10	0	1
5	3	12	1	0
21	3	13	1	0
11	1	11	0	0
23	1	5	1	0
19	3	8	1	1
13	3	4	1	1
22	1	8	1	0
7	3	12	1	0
2	2	12	0	1
7	2	11	1	1
11	3	13	1	0
19	2	7	0	1
19	2	5	1	1
19	3	5	1	1
5	2	15	0	1
29	1	14	1	0
7	1	15	0	0
17	2	3	0	1
19	2	9	1	1

13	3	15	0	0
28	1	5	1	0
12	2	14	1	1
2	2	15	1	1
23	3	10	1	0
16	1	14	0	0
18	2	9	0	1
20	2	8	0	1
6	1	4	0	0
17	1	11	1	0
18	1	11	0	0
30	1	14	0	0
4	1	13	1	1
11	2	10	0	1
10	1	13	0	0
2	1	9	1	1
23	1	14	0	0
23	3	5	1	0
23	1	11	0	0
18	2	8	1	1
29	2	12	0	0
8	2	3	1	1
4	3	5	0	0
8	1	12	1	1
20	3	11	0	0
2	3	8	1	1
5	3	4	0	0
14	1	13	1	0
1	2	8	1	1
19	2	14	1	1
9	1	9	0	0
15	1	8	1	1
9	3	13	1	0
12	3	3	1	1
13	1	1	1	1
24	2	2	0	1
13	3	13	1	0
7	3	10	1	1
23	1	13	1	0
24	2	1	0	1
3	2	8	0	1
26	1	9	1	0
1	1	8	1	1
4	2	6	1	1
9	2	1	1	1
28	1	4	0	0

12	2	2	1	1
26	2	2	1	1
14	2	8	0	1
10	2	8	1	1
26	3	14	0	0
15	3	13	1	0
24	3	3	1	0
2	3	9	0	0
11	1	8	0	1
7	2	12	1	1
13	3	15	1	0
20	1	8	0	1
14	2	10	1	1
12	2	15	1	1
1	1	5	0	0
10	3	11	1	0
24	1	5	0	0
18	2	13	0	1
1	3	14	0	0
9	3	1	0	0
30	1	6	1	0
10	3	2	1	1
18	1	12	1	0
23	1	4	1	0
3	3	8	1	1
13	3	4	0	0
29	1	10	0	0
28	3	1	0	0
22	1	5	1	0
24	2	3	1	1
15	2	13	0	1
14	2	12	0	1
8	2	12	0	1
18	3	15	1	0
23	2	6	1	1
26	3	15	0	0
22	2	10	1	1
25	1	12	1	0
29	3	3	0	0
8	1	9	0	0
19	3	7	1	1
16	3	1	0	0
28	2	3	1	1
18	3	2	1	1
23	1	15	0	0
14	1	14	0	0

10	2	15	1	1
2	1	3	0	0
29	1	15	1	0
11	3	12	1	0
7	1	8	0	0
8	2	7	1	1
22	1	4	0	0
16	3	1	1	1
6	3	11	1	0
25	3	5	1	0
4	1	11	0	0
5	3	7	1	1
7	2	6	0	1
7	3	13	0	0
27	1	12	1	0
14	3	13	0	0
6	3	6	0	0
5	1	2	1	1
24	3	13	1	0
7	3	10	0	0
25	1	10	0	0
11	1	14	0	0
20	1	4	1	1
26	2	3	1	1
12	1	15	0	0
10	2	4	0	1
28	2	8	0	1
13	2	13	0	1
1	3	1	0	0
18	3	14	1	0
29	2	7	1	1
25	3	9	0	0
18	1	9	0	1
5	3	1	1	1
30	1	15	1	0
20	3	14	1	0
11	3	10	0	0
29	3	4	0	0
19	2	6	0	1
24	3	9	0	0
2	3	10	0	0
20	1	9	1	1
21	3	14	1	0
18	2	8	0	1
30	2	6	1	1
23	3	12	1	0

9	3	15	1	0
9	2	8	0	1
18	3	9	0	0
9	2	6	0	1
14	1	6	1	1
2	3	11	1	0
25	1	2	0	0
3	2	14	0	1
14	1	14	1	0
25	2	5	1	1
28	3	2	0	0
25	1	15	1	0
20	1	11	1	0
20	1	3	1	1
11	1	10	0	1
3	2	6	1	1
16	2	5	1	1
15	1	3	0	1
23	2	13	1	0
2	3	2	1	1
6	1	2	1	1
12	2	11	0	1
25	2	2	1	1
29	2	12	1	0
16	2	15	1	1
30	1	8	1	0
21	2	5	0	1
10	1	15	1	1
30	1	10	0	0
5	1	8	1	1
11	2	6	1	1
13	1	8	0	1
14	3	7	1	1
25	3	7	1	0
22	1	3	0	0
4	1	15	0	0
28	1	5	0	0
12	1	4	0	1
27	2	5	1	1
30	3	7	1	0
17	1	13	0	0
5	2	3	0	1
1	3	7	0	0
22	3	1	1	0
20	1	7	0	1
7	3	2	1	1

5	3	8	0	0
16	3	6	1	1
28	3	12	0	0
20	1	12	0	0
25	2	7	1	1
21	1	1	0	0
13	2	15	0	1
15	3	12	1	0
18	1	7	0	1
13	3	5	0	0
24	2	6	1	1
9	2	11	0	1
27	3	1	1	0
4	3	14	0	0
11	2	9	0	1
2	1	15	1	1
16	2	4	1	1
2	1	14	1	1
18	1	5	1	1
19	3	2	1	1
7	3	7	0	0
16	3	3	1	1
9	1	7	0	0
13	3	7	0	0
6	2	6	0	1
24	2	1	1	1
22	1	15	0	0
26	3	7	0	0
10	2	9	1	1
7	2	15	0	1
5	1	12	0	0
9	1	15	1	1
26	1	14	0	0
16	3	2	0	0
29	1	1	1	0
17	1	15	0	0
5	2	4	1	1
9	1	14	0	0
5	3	8	1	1
26	1	11	1	0
27	1	9	1	0
22	1	5	0	0
29	2	14	0	0
3	2	2	0	1
23	1	2	0	0
3	3	7	1	1

9	3	8	1	1
2	1	6	1	1
22	1	6	0	0
13	3	9	1	1
28	2	9	1	1
8	1	13	1	1
3	1	3	1	1
29	2	11	1	0
6	1	12	0	0
25	1	7	0	0
25	3	14	0	0
29	1	5	1	0
3	3	14	1	0
22	1	10	0	0
18	2	6	1	1
17	2	12	0	1
4	3	11	1	0
15	2	7	0	1
3	1	2	1	1
24	3	5	0	0
1	3	6	1	1
24	2	12	1	0
2	2	8	0	1
19	3	12	1	0
19	1	11	1	0
29	3	6	0	0
4	2	5	1	1
7	1	1	1	1
29	1	9	1	0
11	2	12	0	1
10	2	5	0	1
26	1	9	0	0
27	3	6	0	0
8	1	11	0	0
5	3	2	1	1
29	2	8	0	1
6	1	13	0	0
14	2	5	1	1
21	3	1	1	0
6	3	4	0	0
30	2	5	1	1
30	2	10	1	1
7	1	9	1	1
15	1	13	0	0
19	3	4	1	1
14	3	8	0	0

14	3	15	0	0
3	2	10	1	1
7	3	2	0	0
24	2	4	0	1
20	1	10	0	1
2	1	14	0	0
15	1	5	0	1
7	1	6	1	1
10	3	7	0	0
27	3	4	0	0
27	2	1	0	1
7	3	3	0	0
18	3	15	0	0
26	2	6	1	1
14	1	13	0	0
1	3	9	0	0
23	1	8	0	0
16	1	3	1	1
21	1	9	1	0
13	3	2	0	0
24	1	2	1	0
3	1	6	1	1
19	1	14	0	0
10	1	10	0	0
10	3	1	0	0
26	2	1	1	1
24	3	12	1	0
25	1	11	1	0
15	2	2	0	1
8	2	5	1	1
14	3	4	0	0
29	1	6	1	0
11	3	15	0	0
16	2	9	1	1
19	2	12	0	1
1	1	15	0	0
7	1	10	1	1
23	3	11	0	0
19	3	15	0	0
28	2	6	0	1
18	3	9	1	1
10	3	13	0	0
9	3	7	0	0
30	1	9	1	0
5	1	3	1	1
13	1	5	1	1

1	1	1	0	0
19	1	4	0	1
16	2	4	0	1
14	1	11	0	0
30	1	11	0	0
21	1	6	0	0
8	2	2	0	1
11	1	12	1	0
2	3	12	1	0
28	2	7	0	1
20	1	14	0	0
16	1	1	1	1
14	3	13	1	0
7	1	15	1	1
12	1	8	0	1
26	2	7	0	1
8	1	8	0	0
27	2	13	1	0
1	1	10	0	0
23	1	6	0	0
26	1	13	0	0
1	1	4	1	1
24	2	15	1	0
16	1	2	0	1
9	2	14	0	1

Tabla de Testeo:

Tiempo	Tipo de Exactitud	N° Inputs	Estado Previo	Resultado Esperado
15	3	15	0	0
26	3	9	1	0
27	3	14	1	0
4	1	9	1	1
26	3	12	0	0
4	1	6	0	0
6	1	3	1	1
29	3	2	1	0
4	3	11	0	0
9	2	9	0	1
8	1	15	0	0
22	3	13	0	0
21	1	12	0	0
30	1	1	0	0
24	3	8	0	0
14	1	15	0	0
17	3	3	0	0
29	3	13	0	0
28	3	12	1	0
30	2	8	0	1
12	1	10	1	1
18	1	4	1	1
17	2	11	0	1
16	3	15	0	0
15	3	14	1	0
29	1	14	0	0
27	2	9	1	1
24	2	12	0	0
26	3	4	1	0
26	1	12	1	0
28	3	3	0	0
28	3	5	1	0
11	2	13	1	1
30	3	1	0	0
9	2	10	1	1
7	2	15	1	1
2	1	11	0	0
3	2	13	1	1
20	2	8	1	1
25	1	15	0	0

26	3	10	1	0
30	3	11	1	0
22	2	9	1	1
21	3	5	0	0
5	1	4	1	1
23	2	15	0	0
11	1	2	0	1
25	1	6	0	0
22	1	3	1	0
15	1	1	0	1
25	2	11	1	0
22	2	11	1	0
18	2	7	0	1
3	3	10	0	0
13	1	2	0	1
17	3	7	0	0
30	3	10	0	0
11	3	7	0	0
14	1	12	0	0
18	2	3	1	1
1	3	4	1	1
3	3	11	1	0
25	3	4	1	0
6	1	12	1	1
2	1	15	0	0
21	2	11	0	0
28	2	11	0	0
13	3	2	1	1
23	2	12	1	0
13	2	5	0	1
4	3	8	0	0
12	2	13	0	1
26	1	12	0	0
19	2	2	0	1
16	3	5	1	1
30	3	8	1	0
7	1	3	0	0
1	2	3	0	1
6	1	5	0	0
9	1	2	1	1
23	3	7	0	0
3	3	7	0	0
28	3	11	0	0
2	2	10	0	1
25	3	14	1	0
11	2	8	1	1

13	3	12	0	0
8	1	2	1	1
22	3	1	0	0
18	2	12	0	1
4	1	7	1	1
4	2	13	0	1
30	3	5	0	0
17	3	14	1	0
12	3	2	1	1
21	1	2	0	0
20	2	7	1	1
22	2	12	0	0
29	3	7	0	0
18	1	6	0	1
13	1	3	0	1
14	1	9	0	1
12	3	8	1	1
29	1	12	1	0
19	3	13	0	0
18	3	1	0	0
28	2	13	1	0
2	1	4	1	1
28	2	10	1	1
9	1	8	1	1
21	2	11	1	0
19	2	6	1	1
19	3	4	0	0
25	2	4	1	1
2	1	5	1	1
11	1	12	0	0
7	3	15	1	0
11	3	14	0	0
10	1	5	0	0
26	2	4	0	1
29	2	10	0	1
12	2	15	0	1
3	1	4	0	0
8	3	12	1	0
6	3	5	1	1
8	1	8	1	1
4	1	14	0	0
21	2	6	1	1
9	1	1	0	0
26	2	15	0	0
19	1	9	0	1
21	2	7	0	1

18	2	15	0	1
24	2	7	0	1
13	2	14	0	1
19	1	2	0	1
18	3	8	0	0
5	2	1	0	1
21	1	11	1	0
19	1	12	1	0
12	1	14	1	0
25	2	4	0	1
12	3	15	0	0
28	2	13	0	0
9	1	14	1	1
27	1	8	0	0
28	3	15	1	0
18	3	11	1	0
20	3	6	1	1
24	2	10	1	1
25	2	15	1	0
2	3	11	0	0
15	3	7	1	1
5	2	6	0	1
19	2	8	1	1
3	2	9	0	1
14	1	8	0	1
19	3	14	0	0
22	3	15	1	0
17	3	12	0	0
11	3	4	1	1
9	2	13	1	1
4	3	2	0	0
17	1	1	0	1
22	3	5	0	0
18	1	12	0	0
22	3	9	1	0
8	1	5	0	0
19	2	3	0	1
26	2	5	0	1
23	3	11	1	0
1	3	7	1	1
24	1	12	0	0
2	3	15	1	0
9	3	4	1	1
23	1	9	1	0
3	2	12	0	1
3	1	7	1	1

15	3	6	1	1
16	2	9	0	1
12	2	10	0	1
14	1	2	0	1
26	3	9	0	0
28	3	13	1	0
3	1	7	0	0
3	2	13	0	1
16	1	5	0	1
23	3	1	0	0
21	3	10	0	0
8	2	2	1	1
6	3	12	1	0
15	2	8	1	1
15	3	7	0	0
11	1	9	0	1
21	2	3	0	1
18	1	10	1	1
12	3	7	0	0
11	3	11	0	0
20	3	1	0	0
13	1	4	1	1
10	3	2	0	0
28	1	13	0	0
1	3	9	1	1
9	2	9	1	1
23	2	1	0	1
28	3	10	1	0
4	2	2	1	1
8	2	9	0	1
30	1	7	1	0
20	2	14	0	1
15	1	11	0	0
21	1	10	1	0
2	2	9	0	1
23	2	14	1	0
6	1	10	1	1
4	2	4	0	1
12	2	8	1	1
27	2	7	0	1
2	2	8	1	1
18	1	11	1	0
20	1	15	1	0
8	1	9	1	1
26	1	15	0	0
23	3	13	1	0

24	2	3	0	1
13	2	3	1	1
16	2	7	0	1
19	3	11	1	0
16	3	10	1	1
29	1	3	1	0
11	3	3	0	0
28	2	14	1	0
18	2	4	0	1
20	3	12	0	0
26	2	6	0	1
4	3	15	1	0
24	1	13	0	0
13	3	8	1	1
9	3	8	0	0
9	1	3	0	0
2	3	1	0	0
25	1	2	1	0
5	3	14	1	0
28	2	2	0	1
18	2	12	1	1
26	2	5	1	1
26	2	8	1	1
12	1	11	1	0
27	1	3	0	0
22	2	1	1	1
30	1	5	0	0
23	2	11	0	0
30	3	1	1	0
23	3	9	0	0
24	1	4	1	0
17	1	3	0	1
29	3	3	1	0
11	2	8	0	1
29	2	9	1	1
22	2	15	0	0
12	1	12	0	0
25	2	12	0	0
18	1	8	0	1
12	1	4	1	1
11	2	1	0	1
1	3	11	1	0
1	2	11	1	1
12	3	5	1	1
1	1	12	0	0
18	3	7	1	1

10	2	7	0	1
27	1	4	0	0
3	1	8	0	0
9	2	11	1	1
29	3	12	0	0
17	3	2	0	0
28	3	8	0	0
16	2	1	1	1
26	2	13	1	0
27	2	15	1	0
21	3	8	0	0
10	3	5	0	0
14	3	2	0	0
13	2	10	0	1
11	2	3	1	1
27	1	9	0	0
13	3	5	1	1
8	3	6	1	1
21	2	9	1	1
17	1	7	1	1
9	1	3	1	1
6	1	2	0	0
10	2	12	1	1
5	2	2	0	1
1	3	2	0	0
23	3	10	0	0
4	3	3	0	0
19	2	1	1	1
15	1	12	1	0
17	2	7	1	1
29	1	2	1	0
26	3	2	0	0
29	1	6	0	0
19	3	10	0	0
4	3	9	1	1
30	1	2	1	0
12	3	11	1	0
13	1	10	1	1
28	2	15	1	0
21	3	15	1	0
17	3	15	1	0
5	2	12	0	1
17	1	7	0	1
9	1	4	1	1
15	1	9	0	1
26	1	3	0	0

20	1	2	0	1
6	3	9	0	0
8	3	10	1	1
18	1	13	1	0
21	1	10	0	0
11	3	9	0	0
4	1	15	1	1
9	3	12	0	0
24	1	14	0	0
17	1	13	1	0
13	1	1	0	1
19	2	11	1	1
13	2	10	1	1
9	2	12	0	1
13	3	1	1	1
20	2	7	0	1
14	3	3	0	0
3	3	5	1	1
30	2	3	1	1
2	1	2	1	1
15	2	8	0	1
19	3	3	0	0
30	3	14	0	0
22	2	14	1	0
4	1	5	1	1
2	3	3	0	0
21	2	8	0	1
28	2	6	1	1
23	1	1	1	0
22	3	3	1	0
21	3	9	0	0
29	1	12	0	0
1	1	12	1	1
13	3	9	0	0
6	1	5	1	1
21	2	3	1	1
6	2	8	1	1
4	1	1	1	1
6	2	10	1	1
25	2	7	0	1
25	2	1	1	1
14	2	6	0	1
11	2	13	0	1
15	3	15	1	0
2	3	4	1	1
18	2	7	1	1

17	1	4	0	1
8	1	4	0	0
18	3	12	0	0
11	2	14	0	1
26	3	13	0	0
24	1	10	0	0
4	2	11	1	1
18	1	2	0	1
21	2	10	0	1
6	2	1	1	1
7	2	8	0	1
1	2	6	1	1
16	1	15	1	0
26	1	2	1	0
3	2	12	1	1
4	2	3	1	1
13	2	6	1	1
2	2	14	0	1
28	3	11	1	0
29	2	2	0	1
3	2	9	1	1
26	1	6	1	0
27	3	9	1	0
13	1	5	0	1
30	1	8	0	0
13	2	13	1	1
12	3	14	1	0
4	1	6	1	1
6	1	6	0	0
5	1	1	0	0
15	2	2	1	1
20	2	6	1	1
27	3	12	1	0
27	1	10	0	0
27	2	2	0	1
16	2	1	0	1
4	2	10	1	1
13	2	4	1	1
17	3	6	1	1
19	3	13	1	0
15	1	2	0	1
4	3	12	0	0
10	1	7	0	0
13	3	7	1	1
2	3	1	1	1
14	3	9	1	1

5	1	14	1	1
12	1	12	1	0
27	1	15	0	0
11	1	7	1	1
25	3	8	0	0
19	1	15	1	0
8	1	7	1	1
28	3	5	0	0
15	2	11	0	1
16	1	7	0	1
1	2	13	0	1
29	3	4	1	0
22	1	2	0	0
22	3	10	1	0
15	2	7	1	1
27	3	13	1	0
24	3	3	0	0
10	1	5	1	1
17	2	13	1	1
14	2	4	1	1
4	2	13	1	1
1	3	14	1	0
2	1	8	1	1
9	3	14	0	0
7	1	1	0	0
28	3	6	1	0
7	1	7	1	1
7	3	11	0	0
7	2	6	1	1
12	3	9	1	1
1	1	13	0	0
13	2	11	1	1
5	1	15	0	0
20	1	9	0	1
13	2	12	1	1
16	3	5	0	0
3	3	1	1	1
22	3	5	1	0
16	3	4	1	1
16	1	10	1	1
20	2	5	0	1
6	1	13	1	1
1	2	14	0	1
3	1	8	1	1
13	1	12	0	0
4	3	13	1	0

5	3	3	1	1
9	2	6	1	1
8	2	10	0	1
10	3	9	0	0
9	2	4	0	1
28	3	9	1	0
4	2	15	0	1
16	1	2	1	1
27	2	4	0	1
29	2	3	0	1
4	1	4	0	0
9	3	11	1	0
18	3	13	1	0
25	3	15	0	0
3	3	8	0	0
4	3	15	0	0
15	1	15	1	0
24	2	11	1	0
22	1	12	1	0
26	2	2	0	1
20	1	1	0	1
5	1	11	0	0
12	3	15	1	0
11	2	3	0	1
22	2	12	1	0
24	2	6	0	1
14	2	15	0	1
26	2	9	1	1
29	3	7	1	0
9	1	15	0	0
19	3	12	0	0
24	1	15	1	0
7	2	11	0	1
30	3	8	0	0
12	3	8	0	0
21	2	4	1	1
20	2	15	1	1
10	1	6	1	1
16	2	10	1	1
16	2	6	1	1
19	1	3	1	1
8	2	5	0	1
10	3	10	1	1
20	1	4	0	1
16	3	10	0	0
21	1	11	0	0

29	3	5	1	0
10	1	7	1	1
15	2	1	0	1
10	1	2	1	1
10	3	13	1	0
22	1	2	1	0
2	3	15	0	0
8	3	14	0	0
2	3	12	0	0
3	2	11	0	1
26	2	15	1	0
29	1	5	0	0
3	3	4	1	1
10	2	14	0	1
6	2	14	0	1
9	3	11	0	0
16	1	1	0	1
9	3	10	1	1
20	2	3	1	1
16	1	13	1	0
6	2	7	0	1
28	3	10	0	0
15	1	1	1	1
27	3	2	1	0
12	2	6	0	1
24	3	10	0	0
16	1	9	1	1
14	3	5	0	0
26	1	15	1	0
20	3	8	0	0
2	2	4	0	1
9	1	9	1	1
28	1	12	0	0
4	3	3	1	1
9	3	9	0	0
16	2	7	1	1
20	2	5	1	1
3	2	15	0	1
26	3	3	0	0
1	2	1	0	1

Tabla de Validación:

Tiempo	Tipo de Exactitud	N° Inputs	Estado Previo	Resultado Esperado
30	2	15	1	0
26	1	7	0	0
28	3	14	1	0
29	2	10	1	1
2	3	7	0	0
6	1	3	0	0
26	3	4	0	0
16	2	12	0	1
11	2	2	0	1
15	1	4	1	1
18	3	10	1	1
27	2	14	0	0
30	1	3	0	0
9	2	14	1	1
18	3	13	0	0
21	1	15	0	0
27	3	15	1	0
17	3	1	1	1
22	2	4	1	1
12	1	9	0	1
23	2	15	1	0
5	3	15	1	0
1	3	3	0	0
19	3	6	0	0
4	2	2	0	1
30	2	7	1	1
1	1	9	0	0
17	2	6	0	1
6	2	13	0	1
17	1	8	0	1
4	2	15	1	1
24	1	4	0	0
1	2	1	1	1
15	2	11	1	1
10	1	4	0	0
14	1	2	1	1
18	3	4	0	0
6	3	1	1	1
14	2	13	0	1
23	3	3	1	0

25	1	4	1	0
11	3	15	1	0
25	3	15	1	0
10	2	2	0	1
14	2	5	0	1
25	3	6	1	0
19	1	9	1	1
28	1	3	1	0
17	2	13	0	1
28	1	14	0	0
12	2	5	0	1
16	1	13	0	0
8	1	3	1	1
10	3	1	1	1
16	3	2	1	1
20	2	9	1	1
28	1	15	1	0
10	3	15	0	0
21	3	9	1	0
29	1	3	0	0
23	1	12	0	0
28	1	2	1	0
2	2	12	1	1
21	1	5	0	0
2	3	6	0	0
9	2	12	1	1
14	1	3	1	1
4	2	12	0	1
6	2	8	0	1
23	3	3	0	0
23	2	8	1	1
27	1	15	1	0
30	1	12	1	0
25	1	14	1	0
23	3	8	1	0
12	1	1	0	1
1	3	13	1	0
30	2	4	0	1
3	2	8	1	1
16	2	12	1	1
17	3	15	0	0
20	1	13	0	0
4	2	9	0	1
24	3	2	1	0
10	2	4	1	1
20	3	3	0	0

28	3	14	0	0
23	3	15	1	0
22	3	10	0	0
27	2	11	1	0
13	2	11	0	1
22	3	14	1	0
11	1	14	1	0
10	2	1	0	1
5	3	13	1	0
24	1	1	1	0
23	3	13	0	0
1	2	4	1	1
19	3	9	1	1
23	2	7	0	1
1	2	5	1	1
12	3	11	0	0
5	1	15	1	1
20	1	15	0	0
23	3	8	0	0
10	2	8	0	1
16	2	2	1	1
26	2	11	1	0
5	3	10	1	1
23	2	7	1	1
22	3	15	0	0
2	2	11	1	1
21	3	12	1	0
6	3	15	0	0
8	2	10	1	1
30	3	2	1	0
19	2	12	1	1
6	1	8	0	0
13	1	8	1	1
23	2	10	0	1
20	1	13	1	0
20	1	12	1	0
15	1	13	1	0
15	2	13	1	1
4	3	2	1	1
3	3	5	0	0
13	1	3	1	1
17	3	8	1	1
6	1	15	0	0
27	1	7	1	0
29	2	6	0	1
25	2	8	0	1

6	1	14	0	0
24	3	15	0	0
23	3	4	1	0
17	1	2	0	1
14	1	8	1	1
16	1	11	0	0
10	1	13	1	1
23	3	14	1	0
21	2	7	1	1
11	1	8	1	1
2	3	3	1	1
1	1	3	0	0
3	3	6	1	1
10	1	11	1	1
2	2	2	1	1
7	2	9	1	1
19	3	6	1	1
27	1	2	1	0
4	2	1	1	1
2	2	6	1	1
8	1	11	1	1
27	3	1	0	0
18	3	11	0	0
29	2	3	1	1
3	1	5	0	0
20	1	10	1	1
10	1	15	0	0
24	1	5	1	0
12	1	3	1	1
25	2	2	0	1
5	2	4	0	1
29	3	10	0	0
6	1	9	1	1
29	2	13	0	0
18	1	15	0	0
17	3	3	1	1
7	3	6	0	0
25	3	6	0	0
9	3	14	1	0
27	1	6	0	0
11	2	14	1	1
18	3	8	1	1
1	1	7	1	1
16	1	14	1	0
10	2	12	0	1
6	3	5	0	0

8	1	2	0	0
29	3	11	0	0
29	3	6	1	0
26	1	3	1	0
5	3	14	0	0
30	2	9	1	1
11	2	4	1	1
2	3	7	1	1
9	2	3	1	1
6	2	13	1	1
24	1	13	1	0
5	3	10	0	0
20	3	10	1	1
13	2	8	1	1
1	3	15	1	0
12	1	14	0	0
6	3	7	1	1
21	3	4	1	0
11	1	1	0	1
12	1	8	1	1
19	1	7	1	1
1	2	8	0	1
12	1	7	1	1
22	3	8	0	0
17	3	11	1	0
6	1	10	0	0
4	3	10	0	0
9	3	7	1	1
13	2	9	1	1
25	3	2	1	0
11	1	6	0	1
18	1	14	1	0
8	2	11	0	1
17	2	3	1	1
14	2	7	1	1
28	2	14	0	0
28	1	1	0	0
12	3	12	1	0
27	1	5	0	0
8	1	13	0	0
30	2	11	0	0
6	2	3	0	1
2	2	2	0	1
12	2	12	1	1
19	1	8	1	1
5	2	2	1	1

9	3	2	1	1
23	1	3	1	0
19	2	7	1	1
17	1	10	1	1
17	3	8	0	0
20	2	2	0	1
27	2	5	0	1
22	1	4	1	0
27	2	14	1	0
22	3	6	1	0
4	2	9	1	1
10	1	8	0	0
14	1	4	1	1
12	3	6	1	1
22	2	8	0	1
5	2	8	0	1
17	2	1	1	1
20	3	15	0	0
24	3	2	0	0
17	2	14	0	1
17	1	10	0	1
22	2	6	0	1
8	3	6	0	0
16	1	10	0	1
23	2	12	0	0
25	3	12	1	0
22	1	10	1	0
12	2	11	1	1
5	2	5	0	1
25	3	10	0	0
17	3	9	1	1
16	3	14	1	0
7	3	11	1	0
22	1	7	1	0
25	2	14	1	0
28	2	5	0	1
9	3	12	1	0
2	2	1	1	1
10	2	14	1	1
23	2	4	1	1
23	2	13	0	0
3	1	14	1	1
14	2	4	0	1
17	3	13	0	0
17	3	5	0	0
27	2	12	0	0

30	2	15	0	0
17	3	14	0	0
22	3	14	0	0
13	3	14	1	0
12	2	3	0	1
8	3	12	0	0
22	3	4	1	0
8	2	9	1	1
9	1	8	0	0
30	1	10	1	0
10	2	10	1	1
13	1	14	0	0
10	2	9	0	1
18	3	6	1	1
5	2	5	1	1
19	2	13	0	1
19	3	8	0	0
28	1	1	1	0
19	3	2	0	0
24	1	15	0	0
6	3	14	1	0
21	2	12	0	0
20	2	9	0	1
5	2	9	0	1
21	2	14	1	0
15	3	6	0	0
6	1	1	1	1
8	1	6	1	1
14	2	9	1	1
29	1	10	1	0
6	1	15	1	1
3	1	6	0	0
19	2	2	1	1
2	1	3	1	1
15	3	5	0	0
21	3	6	0	0
27	1	11	1	0
15	3	11	0	0
3	2	10	0	1
27	3	2	0	0
8	3	5	0	0
9	1	13	1	1
6	1	9	0	0
25	2	1	0	1
21	1	14	0	0
15	2	14	1	1

22	1	13	1	0
29	3	9	0	0
5	3	11	0	0
30	2	9	0	1
15	2	10	0	1
14	2	1	0	1
17	1	14	1	0
30	3	12	1	0
27	1	3	1	0
13	2	8	0	1
16	3	11	0	0
12	3	10	1	1
11	1	1	1	1
17	3	13	1	0
4	1	2	0	0
19	3	1	0	0
21	3	15	0	0
18	1	1	0	1
30	3	4	0	0
7	3	14	1	0
29	2	14	1	0
19	1	8	0	1
18	1	3	0	1
6	2	2	1	1
2	1	5	0	0
1	1	2	1	1
18	3	3	1	1
15	2	12	0	1
25	3	11	1	0
7	2	13	0	1
19	1	13	1	0
5	3	9	1	1
6	3	8	1	1
27	2	6	1	1
26	3	5	1	0
20	2	4	1	1
16	2	10	0	1
24	3	7	0	0
26	3	1	0	0
20	1	14	1	0
20	3	15	1	0
19	2	4	1	1
17	2	9	0	1
21	1	13	1	0
4	3	9	0	0
20	2	10	0	1

8	2	7	0	1
6	1	1	0	0
14	3	10	0	0
1	2	5	0	1
25	3	8	1	0
12	2	13	1	1
23	2	5	0	1
8	1	14	0	0
14	1	10	0	1
27	3	15	0	0
17	3	9	0	0
11	2	7	1	1
11	3	13	0	0
23	2	10	1	1
5	3	1	0	0
25	2	9	0	1
21	1	3	1	0
12	2	2	0	1
4	2	8	0	1
16	3	7	0	0
20	2	15	0	1
8	2	8	0	1
29	3	13	1	0
15	2	3	1	1
13	2	6	0	1
7	3	1	0	0
29	1	15	0	0
11	2	9	1	1
23	1	11	1	0
24	1	2	0	0
10	1	9	1	1
22	2	11	0	0
20	1	11	0	0
18	2	2	1	1
19	1	5	1	1
25	1	14	0	0
5	3	12	0	0
9	2	5	1	1
22	1	9	0	0
24	1	3	1	0
12	2	4	0	1
18	2	9	1	1
11	3	5	1	1
7	3	8	1	1
29	2	13	1	0
8	1	15	1	1

7	1	13	1	1
4	1	2	1	1
10	3	6	1	1
25	1	3	1	0
15	1	9	1	1
21	2	4	0	1
27	2	11	0	0
29	2	1	1	1
14	3	2	1	1
20	2	12	1	1
13	2	14	1	1
21	2	14	0	0
6	2	2	0	1
11	3	1	1	1
4	3	1	1	1
2	3	2	0	0
7	2	1	1	1
5	2	13	1	1
8	3	15	0	0
12	3	14	0	0
2	2	9	1	1
24	1	1	0	0
11	3	1	0	0
27	3	11	1	0
21	1	13	0	0
5	2	7	0	1
20	1	5	0	1
26	3	3	1	0
4	1	9	0	0
27	3	11	0	0
9	3	13	0	0
13	1	14	1	0
12	3	4	0	0
14	1	1	0	1
26	1	2	0	0
15	3	9	0	0
2	2	4	1	1
2	2	13	0	1
6	2	4	1	1
2	1	10	1	1
4	2	10	0	1
20	2	3	0	1
22	3	13	1	0
18	1	4	0	1
27	2	8	1	1
20	3	2	0	0

22	2	5	0	1
24	2	5	1	1
10	1	4	1	1
23	2	3	0	1
13	1	7	1	1
3	1	4	1	1
19	2	9	0	1
9	3	2	0	0
25	2	5	0	1
6	2	15	0	1
7	1	13	0	0
22	1	12	0	0
15	2	4	1	1
28	2	4	1	1
7	2	14	0	1
16	3	12	0	0
13	2	1	1	1
15	1	14	0	0
25	1	6	1	0
9	3	3	1	1
14	3	14	0	0
10	1	12	0	0
2	1	13	1	1
20	3	4	1	1
27	1	2	0	0
25	3	1	1	0
30	1	1	1	0
8	1	12	0	0
27	1	7	0	0
19	1	12	0	0
8	2	11	1	1
26	1	10	1	0
7	1	4	0	0
30	3	12	0	0
4	1	1	0	0
27	3	6	1	0
23	1	4	0	0
8	1	6	0	0
6	3	6	1	1
2	1	7	1	1
26	3	6	1	0
20	2	13	1	1
24	3	10	1	0
15	3	1	0	0
17	1	12	0	0
8	1	10	1	1

7	2	3	1	1
26	2	10	0	1
17	2	5	1	1
8	3	1	0	0
4	2	14	1	1
23	2	11	1	0
11	2	12	1	1
28	1	9	0	0
30	1	4	0	0
16	1	15	0	0
7	1	5	0	0
1	1	1	1	1
9	3	6	0	0
5	3	2	0	0
16	3	12	1	0
4	2	5	0	1
8	3	3	1	1
22	1	11	1	0
21	2	2	1	1
23	3	9	1	0
2	1	9	0	0
30	1	5	1	0
18	2	11	1	1
24	2	14	1	0
11	1	10	1	1
15	3	3	0	0
6	2	12	1	1
9	2	10	0	1
12	1	13	0	0
24	1	11	1	0
17	1	6	0	1
2	3	14	0	0
22	2	3	0	1
28	1	11	1	0
1	1	14	1	1
15	3	9	1	1
19	1	7	0	1
10	3	15	1	0
9	3	3	0	0
17	2	2	1	1

```
using UnityEngine;
using System.Collections;

public class Player : MonoBehaviour {
	public int lastItemCollected = 0;
	float h;
	public Rigidbody2D rb;
	//Animator anim;
	SpriteRenderer sp;
	int speed = 28;
	public float jumpSpeed;
	float jumpBounce;
	public int jumpCount = 0;
	private GameObject instance;
	public bool isGrounded = true;
	public float axisX = -23;
	// Use this for initialization
	void Start () {
		rb = GetComponent<Rigidbody2D> ();
		//anim = GetComponent<Animator> ();
		jumpBounce = jumpSpeed * 1.5f;
		sp = GetComponent<SpriteRenderer> ();
	}
```

Player.cs

```
// Update is called once per frame
	void Update () {
		h = Input.acceleration.x;
		//h = Input.GetAxis ("Horizontal");
		//anim.SetFloat ("Move", Mathf.Abs(h));
		if (h >= 0) {
			sp.flipX = false;
		}
		if (h < 0) {
			sp.flipX = true;
		}
		if (this.gameObject.transform.position.x <= -30) {
			transform.position = new Vector2 (27, transform.position.y);
		}

		if (this.gameObject.transform.position.x > 30) {
			transform.position = new Vector2 (-27, transform.position.y);
		}

		if (this.gameObject.transform.position.y <= -8) {
			transform.position = new Vector2 (0, 0);
		}
	}
	void FixedUpdate(){
		rb.velocity = new Vector2 (h * speed, rb.velocity.y);
		if(Input.GetTouch(0).phase == TouchPhase.Began && isGrounded == true){
		//if(Input.GetKey(KeyCode.Z) && isGrounded == true){
			rb.AddForce (Vector2.up * jumpSpeed);
```

```
                        //anim.SetBool ("Jump", true);
                        isGrounded = false;
                        jumpCount++;
                        //StartCoroutine(Reset ());
                }
        }
        void OnCollisionEnter2D(Collision2D obj){
                if (obj.gameObject.CompareTag ("Box")) {
                        obj.gameObject.SetActive (false);
                        rb.velocity = new Vector2 (rb.velocity.x, 0);
                        rb.AddForce (Vector2.up * jumpBounce);
                }
        }
}
```

AddTextFromDB.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;
using System.Data;
using Mono.Data.Sqlite;
using System.IO;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class AddTextFromDB : MonoBehaviour {
        dbAccess db;
        public Text wall;
```

```
	private string connectionString;
	string p;
	public List<string> wordsFromDB;
	string textWall;
	// Use this for initialization
	void Start () {
		db = GetComponent<dbAccess>();
		wall.text = "";
		wordsFromDB = new List<string> ();
	}

	public void GetWords(){
		db.OpenDB ("db_Tesis.sqlite");
		wordsFromDB = db.Words (wordsFromDB);
		db.CloseDB ();
}

	public void TextWallset(){
		textWall = "";
		if (wordsFromDB.Count != 0) {
			for (int i = 0; i < wordsFromDB.Count; i++) {
				textWall += wordsFromDB [i] + "\n";
			}
		}
		wall.text = textWall;
		wordsFromDB.Clear ();
	}
}
```

AudioConv.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class AudioConv : MonoBehaviour {

	public AudioClip audio;
	[HideInInspector]
	public string audiostring;
	public string[] audiosplit;
	ImageConvertion imgc;

	// Use this for initialization
	void Start () {
	//	imgc = new ImageConvertion ();
	//	audiostring = imgc.audioToBase64 (audio);
	//	audiosplit = imgc.Split (audiostring, 66);
	}

	// Update is called once per frame
	void Update () {
	}
}
```

BGColor.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;
public class BGColor : MonoBehaviour {


	SpriteRenderer sprt;
	GameObject timer;
	string scene;
	// Use this for initialization
	void Start () {
		sprt = GetComponent<SpriteRenderer> ();
		sprt.color = Color.gray;
		sprt.sprite = Resources.Load<Sprite> ("Loading");
		timer = GameObject.FindGameObjectWithTag ("Timer");
		scene = SceneManager.GetActiveScene ().name;
	}
	// Update is called once per frame
	void Update () {
		if(timer.activeInHierarchy == false){
			sprt.sprite = Resources.Load<Sprite> (scene);
			sprt.color = Color.white;
	}
	}
}
```

BoxSpawn.cs

```
using UnityEngine;
using System.Collections;
public class BoxSpawn : MonoBehaviour {
	public GameObject box;
	float timeRespawn = 2;
	float startTimer = 0;
	// Use this for initialization
	void Start () {
		box.gameObject.SetActive(true);
	}
	// Update is called once per frame
	void Update () {
		if (box.gameObject.activeInHierarchy == false) {
			startTimer += Time.deltaTime;
			if (startTimer >= timeRespawn) {
				startTimer = 0;
				Respawn ();
			}
		}
	}
	void Respawn(){
		box.gameObject.SetActive (true);
	}
}
```

ButtonControl.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
public class ButtonControl : MonoBehaviour {
	List<GameObject> locks;
	NiceSceneTransition nct;
	public static int option = 1;
	// Use this for initialization
	void Start () {
		Cursor.visible = true;
		nct = GameObject.Find
		("NiceSceneTransition").GetComponent<NiceSceneTransition> ();
	}
	// Update is called once per frame
	void Update () {
	}
	public void Starter (){
		option = 1;
		Debug.Log (option);
	}

	public void Mid (){
		option = 2;
		Debug.Log (option);
	}

	public void Advanced (){
		option = 3;
		Debug.Log (option);
	}
	public void Begin(){
		if (Time.timeScale == 0) {
```

```
                Time.timeScale = 1;
                nct.LoadScene ("DificultySelect");
        } else {
                nct.LoadScene ("DificultySelect");
        }
}

public void BacktoMenu(){
        nct.LoadScene ("MENU");
}

public void StartGame(){
        nct.LoadScene ("StageSelect");
}

public void Lvl1(){
        nct.LoadScene ("Lvl1");
}

public void Lvl2(){
                nct.LoadScene ("Lvl2");
}

public void Lvl3(){
                nct.LoadScene ("Lvl3");
}

public void Lvl4(){
                nct.LoadScene ("Lvl4");
```

```
}
public void Lvl5(){
		nct.LoadScene ("Lvl5");
}
public void Lvl6(){
		nct.LoadScene ("Lvl6");
}
public void Lvl7(){
		nct.LoadScene ("Lvl7");
}
public void Lvl8(){
		nct.LoadScene ("Lvl8");
}
public void Lvl9(){
		nct.LoadScene ("Lvl9");
}
public void Lvl10(){
		nct.LoadScene ("Lvl10");
}
public void Exit(){
	Application.Quit ();
}
public void ResetLetterLast(){
	List<char> tempchar = GameObject.Find ("SManager").GetComponent<SManager>
	().answer;
	int tempCount = GameObject.Find ("Player").GetComponent<Player>
	().lastItemCollected;
	float tempAxis = GameObject.Find ("Player").GetComponent<Player> ().axisX;
	tempchar.RemoveAt (tempCount - 1);
```

```
                tempCount--;
                tempAxis = tempAxis - 3;
        }
}
```

ChangeImg.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class ChangeImg : MonoBehaviour {

        SpriteRenderer sprt;
        Object[] sprites;
        // Use this for initialization
        void Start () {
                sprt = gameObject.GetComponent<SpriteRenderer> ();
                sprites = Resources.LoadAll("Selector");
        }
        // Update is called once per frame
        void Update () {
                if (ButtonControl.option == 1) {
                        sprt.sprite = (Sprite)sprites[1];
                } else if (ButtonControl.option == 2) {
                        sprt.sprite = (Sprite)sprites[2];

                } else if (ButtonControl.option == 3) {
                        sprt.sprite = (Sprite)sprites[3];
                }
        }
}
```

Clock.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class Clock : MonoBehaviour {

	float time = 0f;
	public Text timer;
	public int tiempoInt;
	// Use this for initialization
	void Start () {
		gameObject.GetComponent<Clock> ().enabled = true;
	}

	// Update is called once per frame
	void Update () {
		time += Time.deltaTime;
		tiempoInt = (int)time;
		timer.text = "TIME: " + tiempoInt.ToString();
	}
}
```

CodeTemp.cs

```
using System;
using System.Collections;
using System.IO.Compression;
using System.IO;
using System.Collections.Generic;
using System.Linq;
//using UnityEngine;
public class CodeTemp{

    public List<double[]> multiplicacion_matrices(List<double[]> A, List<double[]> B)
    {
        int n, m, p;
        n = A.Count;
        m = B.Count;
        double[] C = B[0];
        p = C.Length;
        List<double[]> O = crear_matriz_zero (n, p);
        double suma;
        Console.WriteLine ("n: " + n + " m: " + m + " p: " + p);
        for (int i = 0; i < n; i++) {
            double[] fil = A [i];
            for (int j = 0; j < p; j++) {
                suma = 0;
                for (int k = 0; k < m; k++) {
                    double[] temp = B [k];
                    suma += fil [k] * temp [j];
                }
```

```
                    double[] fil2 = O [i];
                    fil2 [j] = suma;
                }
            }
            return O;
}
public List<double[]> multiplicacion_matrices_activacion(List<double[]> A, List<double[]>
B)
{
            int n, m, p;
            n = A.Count;
            m = B.Count;
            double[] C = B [0];
            p = C.Length;
            List<double[]> O = crear_matriz_zero (n, p);
            double suma;
            for (int i = 0; i < n; i++) {
                double[] fil = A [i];
                for (int j = 0; j < p; j++) {
                    suma = 0;
                    for (int k = 0; k < m; k++) {
                        double[] temp = B [k];
                        suma += fil [k] * temp [j];
                    }
                    double[] fil2 = O [i];
                    fil2 [j] = activacion_funcion (suma);
                }
            }
            return O;
```

```
}
public double activacion_funcion(double temp)
{
        double r;

        r = 1.0f/(Math.Exp(-temp)+1);
        return r;
}

public List<double[]> crear_matriz(int n, int m)
{
        List<double[]> M = new List<double[]> ();
        double temp;
        for (int i = 0; i < n; i++) {
                double[] fila = new double[m];
                for (int j = 0; j < m; j++) {
                        Random r = new Random ();
                        temp = r.Next (0, 100000);
                        fila [j] = temp;
                }
                M.Add (fila);
        }
        return M;
}
public List<double[]> crear_matriz_zero(int n, int m)
{
        List<double[]> M = new List<double[]> ();

        for(int i=0; i<n; i++)
```

```
        {
            double[] fila = new double[m];
            M.Add (fila);
        }
        return M;
}
public List<double[]> load_csv(string file, int num_cols)
{
        List<Dictionary<string,object>> data = CSVReader.Read (file);

        List<double[]> doubleList = new List<double[]> ();
        for (int i = 0; i < data.Count; i++) {

            double[] tempdc = new double[num_cols];
            for (int j = 0; j < num_cols; j++) {
                string sj = j.ToString();
                string tempString;
                double tempd;
                tempString = data [i] [sj].ToString();
                tempd = Convert.ToDouble (tempString);
                tempdc [j] = tempd;
            }
            doubleList.Add(tempdc);

        }
        return doubleList;
}

public void show_matrix(List<double[]> L)
```

```
{
        double[] T = L [0];
        int m = T.Length;
        int n = L.Count;

        for(int i=0;i<n; i++){
                T = L [i];
                for (int j = 0; j < m; j++) {
                        System.Console.WriteLine (T [j] + " ");
                }
                System.Console.WriteLine ("");
        }
}
public List<double[]> red_neuronal(string w1_path, string w2_path, string test_path)
{
        List<double[]> W1 = load_csv (w1_path, 2 * 4 + 1);
        List<double[]> W2 = load_csv (w2_path, 1);

        List<double[]> table = load_csv (test_path, 4);
        List<double[]> R = new List<double[]> ();

        R.Add (table [0]);

        int n = 1;
        //creamos primera hipotesis
        List<double[]> H = multiplicacion_matrices_activacion(R,W1);
        show_matrix (H);
        //creamos salida
```

```
        List<double[]> O = multiplicacion_matrices_activacion (H, W2);
        for (int i = 0; i < n; i++) {
                double[] tt = O [i];
                Console.WriteLine ("#" + i + " " + tt [0]);
        }
        return O;
}
        public List<double[]> red_neuronalTrue (string w1, string w2, List<double[]>
parametersGame){
        List<double[]> W1 = load_csv (w1, 2 * 4 + 1);
        List<double[]> W2 = load_csv (w2, 1);
        List<double[]> EX = parametersGame;
        List<double[]> R = new List<double[]> ();
        R.Add (EX [0]);
        int n = 1;

        List<double[]> H = multiplicacion_matrices_activacion (R, W1);
        show_matrix (H);

        List<double[]> O = multiplicacion_matrices_activacion(H,W2);
        for (int i = 0; i < n; i++) {
                double[] tt = O [i];
                Console.WriteLine ("#" + i + " " + tt [0]);
        }
        return O;
}
public static void main(string[] args){
        //TODO Auto-generated method stub
        CodeTemp li = new CodeTemp();
```

```
                li.red_neuronal("/Users/Fernando/Desktop/Test/w1.csv",
                        "/Users/Fernando/Desktop/Test/w2.csv",
"/Users/Fernando/Desktop/Test/data.csv"
                );
                Console.Write ("Hello world!!");
        }
```

ContinueVariables.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class ContinueVariables{
        public ContinueVariables(){}
        public double GetTime(int time, int exactTime){
                double result=0;
                if (time < exactTime - 1) {
                        result = -1;

                }

                if (time >= exactTime - 1 && time <= exactTime + 1) {
                        result = 0;
                }
                if (time > exactTime + 1) {
                        result = 1;
                }
                return result;
        }
```

```
public double GetPalabrasCorrectas(List<char> cantLetras, List<char> letrasCorrectas){
        double palabrasCorrectas = 0;
        if (cantLetras.Count < letrasCorrectas.Count) {
                palabrasCorrectas = -1;
        }

        if (cantLetras.Count == letrasCorrectas.Count) {
                        palabrasCorrectas = 0;
        }

        if(cantLetras.Count > letrasCorrectas.Count){
                        palabrasCorrectas = 1;
        }
        return palabrasCorrectas;
}
public double GetJumps(int jumpcount, int expectedJumps){
        double expectedJump = 0;

        if (jumpcount < expectedJumps) {
                expectedJump = -1;
        }

        if (jumpcount == expectedJumps) {
                expectedJump = 0;
        }
        if (jumpcount > expectedJumps) {
                expectedJump = 1;
```

```
            }
            return expectedJump;
        }

        public double previousLvl(double previous){
            double current = 0;

            if (previous == -1) {
                current = -1;
            }

            if (previous == 1) {
                current = 1;
            }
            return current;
        }
}
```

ControllerLetter.cs

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;
using System.IO;
public class ControllerLetter : MonoBehaviour {
        private Cubos cubo;
        public List<char> letter;
        //public GameObject cube;
        public List<GameObject> cubosGenerados;
        List<GameObject> cubosEjemplo;
```

```
GameObject plyr;
GameObject[] spawn;
GameObject[] platforms;
GameObject[] buttons;
GameObject[] winds;
GameObject timer;
GameObject[] fall;
GameObject[] mplat;
GameObject ans;
string alpha = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
char[] al;
public string answer;
public int actTime;
int index = 0;
int index2 = 0;
int lineIndex;
float increment = 1.5f;
public int siz;
//float delay = 1;
// Use this for initialization

void Start () {
        plyr = GameObject.FindGameObjectWithTag ("Player");
        spawn = GameObject.FindGameObjectsWithTag ("Spawn");
        platforms = GameObject.FindGameObjectsWithTag ("Platform");
        fall = GameObject.FindGameObjectsWithTag ("Fall");
        mplat = GameObject.FindGameObjectsWithTag ("Moving");
        buttons = GameObject.FindGameObjectsWithTag ("Button");
        winds = GameObject.FindGameObjectsWithTag ("Winds");
```

```
        al = alpha.ToCharArray ();
        cubo = new Cubos();
        cubosGenerados = new List<GameObject> ();
        cubosEjemplo = new List<GameObject> ();
        letter = GameObject.Find ("WordGet").GetComponent<WordGet> ().letters;

        timer = GameObject.FindGameObjectWithTag ("Timer");

        Invoke("HideBG",0f);
        Invoke ("CreateObj", 1f);
        Invoke ("PopUp", 1f);
}
// Update is called once per frame
void Update () {
        if (letter.Capacity -1 <= 6) {
                actTime = 10;
        } else {
                actTime = 15;
        }
        if(timer.activeInHierarchy == false){
                StartCoroutine (Hide ());
        }
}
void PopUp(){
        siz = GameObject.Find ("WordGet").GetComponent<WordGet> ().lettercount;

        while (index < siz) {
                cubo.ShowCreate (new Vector2((index*3.5f+ 1.5f) -2 ,15),letter[index]);
                increment = increment + 1.2f;
```

```
                index++;
            }
            cubosEjemplo.AddRange(GameObject.FindGameObjectsWithTag ("Show"));
            CreateWord ();
        }
        void CreateObj(){
            int neorandom = Random.Range (0, 2);
            siz = GameObject.Find ("WordGet").GetComponent<WordGet> ().lettercount;

            while (index2 < siz) {
                cubo.Create (new Vector2(index2 - siz + increment,20),letter[index2]);
                increment = increment + 1.5f;
                index2++;
            }
            if (neorandom > 0) {
                for (int rf = 0; rf < neorandom; rf++) {
                    cubo.Create (new Vector2 (index2 - siz + increment, 20), al
[Random.Range (0, al.Length)]);
                }
            }
            cubosGenerados.AddRange(GameObject.FindGameObjectsWithTag ("Letter"));

            foreach (GameObject obj in cubosGenerados) {
                obj.transform.position = new Vector2 (Random.Range (-20, 20), 20);

                obj.SetActive (false);
            }
        }
        void HideBG(){
```

```
        foreach (GameObject bx in spawn) {
                bx.SetActive (false);
        }
        foreach (GameObject px in platforms) {
                px.SetActive (false);
        }
        foreach (GameObject fx in fall) {
                fx.SetActive (false);
        }
        foreach (GameObject m in mplat) {
                m.SetActive (false);
        }
        foreach (GameObject b in buttons) {
                b.SetActive (false);
        }
        foreach (GameObject b in winds) {
                b.SetActive (false);
        }
        plyr.SetActive (false);
}
IEnumerator Hide(){
        yield return new WaitForSeconds (0);
        foreach (GameObject obj in cubosGenerados) {
                if(obj != null)
                obj.SetActive (true);
        }
        foreach(GameObject obj in cubosEjemplo){
                obj.SetActive (false);
```

```
                ans.SetActive (false);
        }
        if (plyr.gameObject != null) {
                plyr.SetActive (true);
        }
        foreach (GameObject bx in spawn) {
                bx.SetActive (true);
        }
        foreach (GameObject px in platforms) {
                px.SetActive (true);
        }
        foreach (GameObject fx in fall) {
                fx.SetActive (true);
        }
        foreach (GameObject m in mplat) {
                m.SetActive (true);
        }
        foreach (GameObject b in buttons) {
                b.SetActive (true);
        }
        foreach (GameObject b in winds) {
                b.SetActive (true);
        }
}
void CreateWord(){
        answer = GameObject.Find ("WordGet").GetComponent<WordGet> ().RandomAns;
        ans = new GameObject ();

                ans.AddComponent<MeshRenderer> ();
```

```
                ans.AddComponent<TextMesh> ();
                ans.GetComponent<TextMesh> ().text = answer;
                ans.GetComponent<TextMesh> ().characterSize = 1.2f;
                ans.GetComponent<TextMesh> ().anchor = TextAnchor.MiddleCenter;
                ans.gameObject.transform.position = new Vector2 (0,1);
}
}
```

CSVReader.cs

```
using UnityEngine;
using System;
using System.Collections;
using System.Collections.Generic;
using System.Text.RegularExpressions;

public class CSVReader
{
	static string SPLIT_RE = @",(?=(?:[^""]*""[^""]*"")*(?![^""]*""))";
	static string LINE_SPLIT_RE = @"\r\n|\n\r|\n|\r";
	static char[] TRIM_CHARS = { '\"' };

	public static List<Dictionary<string, object>> Read(string file)
	{
		var list = new List<Dictionary<string, object>>();
		TextAsset data = Resources.Load (file) as TextAsset;

		var lines = Regex.Split (data.text, LINE_SPLIT_RE);

		if(lines.Length <= 1) return list;

		var header = Regex.Split(lines[0], SPLIT_RE);
		for(var i=1; i < lines.Length; i++) {

			var values = Regex.Split(lines[i], SPLIT_RE);
			if(values.Length == 0 ||values[0] == "") continue;

			var entry = new Dictionary<string, object>();
```

```
                    for(var j=0; j < header.Length && j < values.Length; j++ ) {
                        string value = values[j];
                        value =
value.TrimStart(TRIM_CHARS).TrimEnd(TRIM_CHARS).Replace("\\", "");
                        object finalvalue = value;
                        int n;
                        float f;
                        if(int.TryParse(value, out n)) {
                            finalvalue = n;
                        } else if (float.TryParse(value, out f)) {
                            finalvalue = f;
                        }
                        entry[header[j]] = finalvalue;
                    }
                    list.Add (entry);
                }
                return list;
        }
}
```

Cubos.cs

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class Cubos {
	private List<Sprite> splist;
	private BoxCollider2D bxc2D;
	private SpriteRenderer sprt;
	public Animator anim;
	private Vector2 _target;
	public char chr;
	public Vector2 target {
		get{ return _target; }
		set{ _target = target; }
	}

	public Cubos(){

	}


	public void Create(Vector2 v2, char c){
		GameObject obj = new GameObject ();
		obj.gameObject.tag = "Letter";
		chr = c;
		string s = chr.ToString ();
		obj.gameObject.name = s;
		obj.gameObject.transform.position = v2;
```

```
obj.transform.localScale += new Vector3 (0.5f, 0.5f, 0);
bxc2D = obj.AddComponent<BoxCollider2D>() as BoxCollider2D;
sprt = obj.AddComponent<SpriteRenderer> () as SpriteRenderer;
anim = obj.AddComponent<Animator> () as Animator;
switch (s) {
case "A":
        sprt.sprite = Resources.Load ("A", typeof(Sprite)) as Sprite;
        anim.runtimeAnimatorController = Resources.Load
("A",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;
        ;
        break;

case "B":
        sprt.sprite = Resources.Load ("B", typeof(Sprite)) as Sprite;
        anim.runtimeAnimatorController = Resources.Load
("B",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;
        anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;
        ;
        break;

case "C":
        sprt.sprite = Resources.Load ("C", typeof(Sprite)) as Sprite;
        anim.runtimeAnimatorController = Resources.Load
("C",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;
        anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;
        ;
        break;

case "D":
        sprt.sprite = Resources.Load ("D", typeof(Sprite)) as Sprite;
```

```
                anim.runtimeAnimatorController = Resources.Load
("D",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "E":

                sprt.sprite = Resources.Load ("E", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("E",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "F":

                sprt.sprite = Resources.Load ("F", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("F",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "G":

                sprt.sprite = Resources.Load ("G", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("G",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;
```

```
            case "H":

                sprt.sprite = Resources.Load ("H", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("H",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "I":

                sprt.sprite = Resources.Load ("I", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("I",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "J":

                sprt.sprite = Resources.Load ("J", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("J",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "K":

                sprt.sprite = Resources.Load ("K", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("K",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;
```

```
                break;


            case "L":

                sprt.sprite = Resources.Load ("L", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("L",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "M":

                sprt.sprite = Resources.Load ("M", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("M",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "N":

                sprt.sprite = Resources.Load ("N", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("N",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "O":

                sprt.sprite = Resources.Load ("O", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("O",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;
```

```
                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "P":

                sprt.sprite = Resources.Load ("P", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("P",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "Q":

                sprt.sprite = Resources.Load ("Q", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("Q",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "R":

                sprt.sprite = Resources.Load ("R", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("R",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "S":

                sprt.sprite = Resources.Load ("S", typeof(Sprite)) as Sprite;
```

```
                anim.runtimeAnimatorController = Resources.Load
("S",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "T":

                sprt.sprite = Resources.Load ("T", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("T",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "U":

                sprt.sprite = Resources.Load ("U", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("U",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


        case "V":

                sprt.sprite = Resources.Load ("V", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("V",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;
```

```
            case "W":

                sprt.sprite = Resources.Load ("W", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("W",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "X":

                sprt.sprite = Resources.Load ("X", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("X",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "Y":

                sprt.sprite = Resources.Load ("Y", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("Y",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;

                break;


            case "Z":

                sprt.sprite = Resources.Load ("Z", typeof(Sprite)) as Sprite;

                anim.runtimeAnimatorController = Resources.Load
("Z",typeof(RuntimeAnimatorController)) as RuntimeAnimatorController;

                anim.cullingMode = AnimatorCullingMode.AlwaysAnimate;

                ;
```

```
                break;
        }
        bxc2D.isTrigger = true;
        obj.AddComponent<LetterBehavior> ();

}
public void ShowCreate(Vector2 vc2, char c){
        GameObject obx = new GameObject ();
        chr = c;
        string s = chr.ToString ();
        obx.gameObject.tag = "Show";
        obx.gameObject.transform.position = vc2;
        obx.gameObject.name = s;
        obx.transform.localScale += new Vector3 (0.7f, 0.7f, 0f);
        sprt = obx.AddComponent<SpriteRenderer> () as SpriteRenderer;
        switch (s) {
        case "A":
                sprt.sprite = Resources.Load ("Letter_A", typeof(Sprite)) as Sprite;
                ;
                break;

        case "B":
                sprt.sprite = Resources.Load ("Letter_B", typeof(Sprite)) as Sprite;
                ;
                break;

        case "C":
                sprt.sprite = Resources.Load ("Letter_C", typeof(Sprite)) as Sprite;
                ;
```

```
            break;

case "D":
            sprt.sprite = Resources.Load ("Letter_D", typeof(Sprite)) as Sprite;
            ;
            break;

case "E":
            sprt.sprite = Resources.Load ("Letter_E", typeof(Sprite)) as Sprite;
            ;
            break;

case "F":
            sprt.sprite = Resources.Load ("Letter_F", typeof(Sprite)) as Sprite;
            ;
            break;

case "G":
            sprt.sprite = Resources.Load ("Letter_G", typeof(Sprite)) as Sprite;
            ;
            break;

case "H":
            sprt.sprite = Resources.Load ("Letter_H", typeof(Sprite)) as Sprite;
            ;
            break;

case "I":
            sprt.sprite = Resources.Load ("Letter_I", typeof(Sprite)) as Sprite;
```

```
        ;
        break;

case "J":
        sprt.sprite = Resources.Load ("Letter_J", typeof(Sprite)) as Sprite;
        ;
        break;

case "K":
        sprt.sprite = Resources.Load ("Letter_K", typeof(Sprite)) as Sprite;
        ;
        break;

case "L":
        sprt.sprite = Resources.Load ("Letter_L", typeof(Sprite)) as Sprite;
        ;
        break;

case "M":
        sprt.sprite = Resources.Load ("Letter_M", typeof(Sprite)) as Sprite;
        ;
        break;

case "N":
        sprt.sprite = Resources.Load ("Letter_N", typeof(Sprite)) as Sprite;
        ;
        break;

case "O":
```

```
            sprt.sprite = Resources.Load ("Letter_O", typeof(Sprite)) as Sprite;
            ;
            break;

    case "P":
            sprt.sprite = Resources.Load ("Letter_P", typeof(Sprite)) as Sprite;
            ;
            break;

    case "Q":
            sprt.sprite = Resources.Load ("Letter_Q", typeof(Sprite)) as Sprite;
            ;
            break;

    case "R":
            sprt.sprite = Resources.Load ("Letter_R", typeof(Sprite)) as Sprite;
            ;
            break;

    case "S":
            sprt.sprite = Resources.Load ("Letter_S", typeof(Sprite)) as Sprite;
            ;
            break;

    case "T":
            sprt.sprite = Resources.Load ("Letter_T", typeof(Sprite)) as Sprite;
            ;
            break;
```

```
case "U":
        sprt.sprite = Resources.Load ("Letter_U", typeof(Sprite)) as Sprite;
        ;
        break;

case "V":
        sprt.sprite = Resources.Load ("Letter_V", typeof(Sprite)) as Sprite;
        ;
        break;

case "W":
        sprt.sprite = Resources.Load ("Letter_W", typeof(Sprite)) as Sprite;
        ;
        break;

case "X":
        sprt.sprite = Resources.Load ("Letter_X", typeof(Sprite)) as Sprite;
        ;
        break;

case "Y":
        sprt.sprite = Resources.Load ("Letter_Y", typeof(Sprite)) as Sprite;
        ;
        break;

case "Z":
        sprt.sprite = Resources.Load ("Letter_Z", typeof(Sprite)) as Sprite;
        ;
        break;
```

```
        }
        //Color cubeColor = new Color (Random.value, Random.value, Random.value, 1f);
        sprt.color = Color.blue;
}

public void CollectedCreate(Vector2 vc2, string s, int c){
        GameObject ob = new GameObject ();
        ob.gameObject.tag = "Collected";
        ob.gameObject.transform.position = vc2;
        ob.gameObject.name = s;
        sprt = ob.AddComponent<SpriteRenderer> () as SpriteRenderer;
        switch (s) {
        case "A":
                sprt.sprite = Resources.Load ("Letter_A", typeof(Sprite)) as Sprite;
                ;
                break;

        case "B":
                sprt.sprite = Resources.Load ("Letter_B", typeof(Sprite)) as Sprite;
                ;
                break;

        case "C":
                sprt.sprite = Resources.Load ("Letter_C", typeof(Sprite)) as Sprite;
                ;
                break;

        case "D":
                sprt.sprite = Resources.Load ("Letter_D", typeof(Sprite)) as Sprite;
```

```
        ;
        break;

case "E":
        sprt.sprite = Resources.Load ("Letter_E", typeof(Sprite)) as Sprite;
        ;
        break;

case "F":
        sprt.sprite = Resources.Load ("Letter_F", typeof(Sprite)) as Sprite;
        ;
        break;

case "G":
        sprt.sprite = Resources.Load ("Letter_G", typeof(Sprite)) as Sprite;
        ;
        break;

case "H":
        sprt.sprite = Resources.Load ("Letter_H", typeof(Sprite)) as Sprite;
        ;
        break;

case "I":
        sprt.sprite = Resources.Load ("Letter_I", typeof(Sprite)) as Sprite;
        ;
        break;

case "J":
```

```
            sprt.sprite = Resources.Load ("Letter_J", typeof(Sprite)) as Sprite;
            ;
            break;

case "K":
            sprt.sprite = Resources.Load ("Letter_K", typeof(Sprite)) as Sprite;
            ;
            break;

case "L":
            sprt.sprite = Resources.Load ("Letter_L", typeof(Sprite)) as Sprite;
            ;
            break;

case "M":
            sprt.sprite = Resources.Load ("Letter_M", typeof(Sprite)) as Sprite;
            ;
            break;

case "N":
            sprt.sprite = Resources.Load ("Letter_N", typeof(Sprite)) as Sprite;
            ;
            break;

case "O":
            sprt.sprite = Resources.Load ("Letter_O", typeof(Sprite)) as Sprite;
            ;
            break;
```

```
case "P":
        sprt.sprite = Resources.Load ("Letter_P", typeof(Sprite)) as Sprite;
        ;
        break;

case "Q":
        sprt.sprite = Resources.Load ("Letter_Q", typeof(Sprite)) as Sprite;
        ;
        break;

case "R":
        sprt.sprite = Resources.Load ("Letter_R", typeof(Sprite)) as Sprite;
        ;
        break;

case "S":
        sprt.sprite = Resources.Load ("Letter_S", typeof(Sprite)) as Sprite;
        ;
        break;

case "T":
        sprt.sprite = Resources.Load ("Letter_T", typeof(Sprite)) as Sprite;
        ;
        break;

case "U":
        sprt.sprite = Resources.Load ("Letter_U", typeof(Sprite)) as Sprite;
        ;
        break;
```

```
case "V":
        sprt.sprite = Resources.Load ("Letter_V", typeof(Sprite)) as Sprite;
        ;
        break;

case "W":
        sprt.sprite = Resources.Load ("Letter_W", typeof(Sprite)) as Sprite;
        ;
        break;

case "X":
        sprt.sprite = Resources.Load ("Letter_X", typeof(Sprite)) as Sprite;
        ;
        break;

case "Y":
        sprt.sprite = Resources.Load ("Letter_Y", typeof(Sprite)) as Sprite;
        ;
        break;

case "Z":
        sprt.sprite = Resources.Load ("Letter_Z", typeof(Sprite)) as Sprite;
        ;
        break;
}
if (c == 0) {
        sprt.color = Color.green;
} else if (c == 1) {
```

```
                    sprt.color = Color.green;
                }
            }

}
```

DataDB.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

public class DataDB {
        //public int rowid{ get; set; }
        public string ENG{ get; set; }
        public string SPA{ get; set; }
        public string SOUND {get;set;}
        public string IMG{ get; set;}
        public string ANS{ get; set; }

        public DataDB(string eng, string spa, string sound, string img, string ans){
        //      this.rowid = rI;
                this.ENG = eng;
                this.SPA = spa;
                this.SOUND = sound;
                this.IMG = img;
                this.ANS = ans;

        }
}
```

dbAccess.cs

```
using UnityEngine;
using System;
using System.IO;
using System.Collections;
using System.Data;
using System.Text;
using Mono.Data.SqliteClient;
using System.Collections.Generic;

public class dbAccess : MonoBehaviour {
	private string connection;
	private IDbConnection dbcon;
	private IDbCommand dbcmd;
	private IDataReader reader;
	private StringBuilder builder;
	public List<DataDB> dataDB = new List<DataDB> ();
	// Use this for initialization
	void Start () {
	}

	public void OpenDB(string p)
	{
		Debug.Log("Call to OpenDB:" + p);
		// check if file exists in Application.persistentDataPath
		string filepath = Application.persistentDataPath + "/" + p;
		if(!File.Exists(filepath))
		{
```

```
                Debug.LogWarning("File \"" + filepath + "\" does not exist. Attempting to
create from \"" +
                        Application.dataPath + "!/assets/" + p);
                // if it doesn't ->
                // open StreamingAssets directory and load the db ->
                WWW loadDB = new WWW("jar:file://" + Application.dataPath + "!/assets/"
+ p);
                while(!loadDB.isDone) {}
                // then save to Application.persistentDataPath
                File.WriteAllBytes(filepath, loadDB.bytes);
            }

            //open db connection
            connection = "URI=file:" + filepath;
            Debug.Log("Stablishing connection to: " + connection);
            dbcon = new SqliteConnection(connection);
            dbcon.Open();
        }

        public void CloseDB(){
            reader.Close(); // clean everything up
            reader = null;
            dbcmd.Dispose();
            dbcmd = null;
            dbcon.Close();
            dbcon = null;
        }

        public IDataReader BasicQuery(string query){ // run a basic Sqlite query
```

```
            dbcmd = dbcon.CreateCommand(); // create empty command
            dbcmd.CommandText = query; // fill the command
            reader = dbcmd.ExecuteReader(); // execute command which returns a reader
            return reader; // return the reader

    }

    public bool CreateTable(string name,string[] col, string[] colType){ // Create a table, name,
column array, column type array
            string query;
            query  = "CREATE TABLE " + name + "(" + col[0] + " " + colType[0];
            for(var i=1; i< col.Length; i++){
                    query += ", " + col[i] + " " + colType[i];
            }
            query += ")";
            try{
                    dbcmd = dbcon.CreateCommand(); // create empty command
                    dbcmd.CommandText = query; // fill the command
                    reader = dbcmd.ExecuteReader(); // execute command which returns a reader
            }
            catch(Exception e){

                    Debug.Log(e);
                    return false;
            }
            return true;
    }

    public int InsertIntoSingle(string tableName, string colName , string value ){ // single insert
```

```
            string query;
            query = "INSERT INTO " + tableName + "(" + colName + ") " + "VALUES (" + value
+ ")";
            try
            {
                  dbcmd = dbcon.CreateCommand(); // create empty command
                  dbcmd.CommandText = query; // fill the command
                  reader = dbcmd.ExecuteReader(); // execute command which returns a reader
            }
            catch(Exception e){

                  Debug.Log(e);
                  return 0;
            }
            return 1;
      }

      public void LoadData(string currentLvl, List<DataDB> ldb){
            ldb.Clear ();
            string query;
            query = "SELECT * FROM " + currentLvl;
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
            while (reader.Read ()) {
                  ldb.Add (new DataDB (reader.GetString (0), reader.GetString (1),
reader.GetString (2), reader.GetString (3), reader.GetString (4)));
            }
      }
```

```
        public void CreateCopyDB(){

                string query;

                query = "CREATE TABLE 'COPY_STARTER_DECK' ('ID' INTEGER PRIMARY
KEY  NOT NULL , 'ENG' TEXT NOT NULL  UNIQUE , 'SPA' TEXT NOT NULL  UNIQUE ,
'SOUND' BLOB NOT NULL  UNIQUE , 'IMG' BLOB NOT NULL  UNIQUE , 'ANS' TEXT NOT
NULL  UNIQUE, 'CL_ID' INTEGER, FOREIGN KEY (CL_ID) REFERENCES CLASES(ID))";

                dbcmd = dbcon.CreateCommand();

                dbcmd.CommandText = query;

                reader = dbcmd.ExecuteReader();

        }


        public void FillCopyDB(){

                string query;

                query = "INSERT INTO COPY_STARTER_DECK SELECT * FROM
STARTER_DECK WHERE CL_ID = " + ButtonControl.option;

                dbcmd = dbcon.CreateCommand();

                dbcmd.CommandText = query;

                reader = dbcmd.ExecuteReader();

        }


        public void Select10(string lv){

                string query;

                query = "CREATE TABLE " + lv + " AS SELECT ENG,SPA,SOUND,IMG,ANS
FROM COPY_STARTER_DECK ORDER BY RANDOM() LIMIT 10";

                dbcmd = dbcon.CreateCommand();

                dbcmd.CommandText = query;

                reader = dbcmd.ExecuteReader();

        }


        public void Delete10(string lv){
```

```
            string query;
            query = "DELETE FROM COPY_STARTER_DECK WHERE ENG IN (SELECT
ENG FROM " + lv + ")";
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
        }

        public List<string> Words(List<string> words){
            string query;
            query = "SELECT ENG FROM STARTER_DECK WHERE CL_ID = " +
ButtonControl.option;
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
            while (reader.Read ()) {
                words.Add(reader.GetString(0));
            }
            return words;
        }
        public void KillClone(){
            string query;
            query = "DROP TABLE COPY_STARTER_DECK";
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
        }

        public void EmptyTableLvl(string lv){
```

```
            string query;
            query = "DROP TABLE " + lv;
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
        }
        public int InsertIntoSpecific(string tableName, string[] col, string[] values){ // Specific insert with col and values
            string query;
            query = "INSERT INTO " + tableName + "(" + col[0];
            for(int i=1; i< col.Length; i++){
                query += ", " + col[i];
            }
            query += ") VALUES (" + values[0];
            for(int i=1; i< col.Length; i++){
                query += ", " + values[i];
            }
            query += ")";
            Debug.Log(query);
            try
            {
                dbcmd = dbcon.CreateCommand();
                dbcmd.CommandText = query;
                reader = dbcmd.ExecuteReader();
            }
            catch(Exception e){

                Debug.Log(e);
                return 0;
```

```
            }
            return 1;
        }

        public int InsertInto(string tableName , string[] values ){ // basic Insert with just values
            string query;
            query = "INSERT INTO " + tableName + " VALUES (" + values[0];
            for(int i=1; i< values.Length; i++){
                query += ", " + values[i];
            }
            query += ")";
            try
            {
                dbcmd = dbcon.CreateCommand();
                dbcmd.CommandText = query;
                reader = dbcmd.ExecuteReader();
            }
            catch(Exception e){

                Debug.Log(e);
                return 0;
            }
            return 1;
        }

        public ArrayList SingleSelectWhere(string tableName , string itemToSelect,string wCol,string
wPar, string wValue){ // Selects a single Item
            string query;
```

```
            query = "SELECT " + itemToSelect + " FROM " + tableName + " WHERE " + wCol
+ wPar + wValue;
            dbcmd = dbcon.CreateCommand();
            dbcmd.CommandText = query;
            reader = dbcmd.ExecuteReader();
            //string[,] readArray = new string[reader, reader.FieldCount];
            string[] row = new string[reader.FieldCount];
            ArrayList readArray = new ArrayList();
            while(reader.Read()){
                  int j=0;
                  while(j < reader.FieldCount)
                  {
                        row[j] = reader.GetString(j);
                        j++;
                  }
                  readArray.Add(row);
            }
            return readArray; // return matches
      }

      // Update is called once per frame
      void Update () {

      }
}
```

FallPlatform.cs

```
using UnityEngine;
using System.Collections;
public class FallPlatform : MonoBehaviour {

	Rigidbody2D rb;
	public float fallDelay;
	// Use this for initialization
	void Start () {
		rb = GetComponent<Rigidbody2D> ();
	}

	// Update is called once per frame
	void Update () {
	}
	void OnCollisionEnter2D(Collision2D obj){
		if (obj.gameObject.CompareTag ("Player")) {
			StartCoroutine (falling ());
		}
	}
		IEnumerator falling(){
			yield return new WaitForSeconds(fallDelay);
			rb.isKinematic = false;
			this.gameObject.GetComponent<BoxCollider2D> ().isTrigger = true;
			yield return new WaitForSeconds(fallDelay + 5);
			this.gameObject.SetActive (false);
```

```
                yield return 0;
            }
}
```

GroundDetect.cs

```
using UnityEngine;
using System.Collections;
public class GroundDetect : MonoBehaviour {
    Rigidbody2D rb;

    // Use this for initialization
    void Start () {
    }

    // Update is called once per frame
    void Update () {
    }

    void OnTriggerEnter2D(Collider2D obj){
        if(obj.gameObject.CompareTag("Platform")){
            this.gameObject.GetComponentInParent<Player> ().isGrounded = true;
        }

        if(obj.gameObject.CompareTag("F.Platform")){
            this.gameObject.GetComponentInParent<Player> ().isGrounded = true;
        }

        if (obj.gameObject.CompareTag("M.Platform")) {
            this.gameObject.GetComponentInParent<Player> ().isGrounded = true;
```

```
                gameObject.GetComponentInParent<Player> ().transform.parent =
                obj.transform;
        }
    }

    void OnTriggerExit2D (Collider2D obj){
        if (obj.gameObject.CompareTag("M.Platform")) {
            gameObject.GetComponentInParent<Player> ().transform.parent = null;
        }
    }
}
```

Hourglass.cs

```
using UnityEngine;
using System.Collections;
public class Hourglass : MonoBehaviour {
    GameObject timer;
    // Use this for initialization
    void Start () {
        timer = GameObject.FindGameObjectWithTag ("Timer");

    }

    // Update is called once per frame
    void Update () {
        if(timer.activeInHierarchy == false){
            Destroy (this.gameObject);
        }
    }
```

```
}
```

ImageConvertion.cs

```
using UnityEngine;
#if UNITY_EDITOR
using UnityEditor;
#endif
using System;
using System.Drawing;
using System.IO;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
        /// <summary>
        /// Description of ImageConverter.
        /// </summary>
        public class ImageConvertion
        {
                public ImageConvertion()
                {
                }
        public string imageToBase64(Texture2D tex)
                {
                        byte[] bytes = tex.EncodeToPNG ();
                        String s = Convert.ToBase64String (bytes);
                        return s;
                }
```

```
public Sprite Base64ToImage(string s)
        {
        Texture2D tex = new Texture2D (2,2);
        byte[] bytes = Convert.FromBase64String (s);
        tex.LoadImage (bytes);
        Sprite sprt = Sprite.Create (tex, new Rect (0, 0, tex.width, tex.height), new Vector2
(0.5f, 0.5f));
        return sprt;
        }

public AudioClip Base64ToAudio(string s, string name){
        byte[] bytes = Convert.FromBase64String (s);
        Debug.Log (bytes.Length);
        float[] fl = ConvertByteToFloat (bytes);
        AudioClip audio = AudioClip.Create (name, fl.Length, 1, 44100, false);
        audio.SetData (fl, 0);
        return audio;
}
private static float[] ConvertByteToFloat(byte[] array)
{
        float[] floatArr = new float[array.Length / 2];
        //float[] floatArr = new float[array.Length / 4];
        for (int i = 0; i < floatArr.Length; i++)
        {
                //if (BitConverter.IsLittleEndian)
                //        Array.Reverse (array, i * 4, 4);
                floatArr[i] = ((float)BitConverter.ToInt16(array, i * 2))/32768;
                //floatArr[i] = BitConverter.ToSingle(array,i*4) / 0x80000000;
```

```
            }
            //Debug.Log (floatArr.Length);
            return floatArr;

    }

    public String[] Split(string text, int charCount){
            if (text.Length == 0)
                    return new string[0];
            int arrayLength = (text.Length - 1) / charCount + 1;
            string[] result = new string[arrayLength];
            for (int i = 0; i < arrayLength - 1; i++) {
                    result [i] = text.Substring (i * charCount, charCount);
            }
            result [arrayLength - 1] = text.Substring ((arrayLength - 1) * charCount);
            return result;
            }

    public static void Main()
            {
            }
    }
```

ImgConvert.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ImgConvert : MonoBehaviour {
```

```
        ImageConvertion imgc;
        public Sprite img;
        [HideInInspector]
        public string imgString;
        Texture2D txt2d;
        PNGtoTexture2D ptt;
        // Use this for initialization
        void Start () {
                imgc = new ImageConvertion ();
                ptt = new PNGtoTexture2D ();
                txt2d = ptt.ConvertSprite (img);
                imgString = imgc.imageToBase64 (txt2d);
        }
        // Update is called once per frame
        void Update () {
        }
}
```

LerpPlatform.cs

```
using UnityEngine;
using System.Collections;

public class LerpPlatform : MonoBehaviour {

        public Vector2 posStart;
        public Vector2 posEnd;
        public float duracion = 1;
        private float lerp = 0;
```

```
	// Use this for initialization
	void Start () {
		transform.localPosition = posStart;
	}

	// Update is called once per frame
	void Update () {
		lerp = Mathf.PingPong (Time.time, duracion) / duracion;
		transform.localPosition = Vector2.Lerp (posStart, posEnd, lerp);
	}

	void OnCollisionEnter2D(Collision2D obj){
		if (obj.gameObject.CompareTag("Player")) {
			obj.gameObject.GetComponent<Player> ().transform.parent =
this.gameObject.transform;
		}
	}

	void OnCollisionExit2D(Collision2D obj){
		obj.gameObject.GetComponent<Player> ().transform.parent = null;
	}
}
```

LetterBehavior.cs

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;
using System.IO;
```

```
public class LetterBehavior : MonoBehaviour {
	private Cubos cubo;
	List<char> letters;
	char[] tempResul = new char[15];
	List<char> resultado;
	[HideInInspector]
	public List<GameObject> cubosgen;
	// Use this for initialization
	void Start () {
		cubosgen = new List<GameObject> ();
		resultado = GameObject.Find ("SManager").GetComponent<SManager> ().answer;
		cubo = new Cubos ();

		letters = GameObject.Find ("WordGet").GetComponent<WordGet> ().letters;
		for (int i = 0; i < letters.Count; i++) {
			tempResul [i] = letters [i];
		}
	}
	// Update is called once per frame
	void Update () {

	}

	void OnTriggerEnter2D(Collider2D obj){
		if (obj.gameObject.CompareTag ("Player")) {
			string s = tempResul [obj.gameObject.GetComponent<Player>
().lastItemCollected].ToString();
			if (this.gameObject.name.Equals (s)) {
				resultado.Add (s [0]);
```

```
                                obj.gameObject.GetComponent<Player> ().lastItemCollected++;

                                Destroy (this.gameObject);


                                cubo.CollectedCreate (new Vector2
(obj.gameObject.GetComponent<Player> ().axisX, -2), s, 0);

                                obj.gameObject.GetComponent<Player> ().axisX =
obj.gameObject.GetComponent<Player> ().axisX + 3;


                        } else {

                                resultado.Add (this.gameObject.name[0]);

                                Destroy (this.gameObject);

                                obj.gameObject.GetComponent<Player> ().lastItemCollected++;

                                Debug.Log ("Error");

                                cubo.CollectedCreate (new Vector2
(obj.gameObject.GetComponent<Player> ().axisX, -2), this.gameObject.name, 1);

                                obj.gameObject.GetComponent<Player> ().axisX =
obj.gameObject.GetComponent<Player> ().axisX + 3;

                        }

                }

        }

        void OnDestroy(){

                Instantiate (Resources.Load ("Boom",
typeof(GameObject)),transform.position,transform.rotation);

        }

}
```

LetterControl.cs

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;
public class LetterControl : MonoBehaviour {
	public List<Vector2> posiciones;
	//public Vector2[] posiciones = new Vector2[20];
	public float xp;
	public float yp;
	float offsetx = 0.5f;
	float offsety = 0.5f;
	// Use this for initialization
	void Start () {
		posiciones = new List<Vector2> ();
		RandomPos ();
	}
	// Update is called once per frame
	void Update () {
	}
		void RandomPos(){
		for(int i = 0; i < 40; i++){
			Vector2 pos = new Vector2 (Random.Range(-19,19), Random.Range (5, 20));
			while (posiciones.FindIndex(delegate(Vector2 p) {
				return p == pos;
			}) != -1){
				pos = new Vector2 (Random.Range(-19,19) + offsetx, Random.Range
(5, 20) + offsety);
			}
			posiciones.Add (pos);}}}
```

LevelsContent.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class LevelsContent : MonoBehaviour {

	SelectClass sc;
	// Use this for initialization
	void Start () {
		sc = GameObject.Find ("ClassDBSelect").GetComponent<SelectClass> ();
		sc.ContentLevels ();
	}
	// Update is called once per frame
	void Update () {
	}
}
```

LockSystem.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class LockSystem : MonoBehaviour {
	public GameObject[] locks;
	public string[] levelLocks;
	public bool[] levelUnlock;
	public Button[] levels;
	// Use this for initialization
```

```
    void Start () {
        for (int i = 0; i < levelLocks.Length; i++)
        {
            if (PlayerPrefs.GetInt (levelLocks [i]) == null) {
                levelUnlock [i] = false;
            } else if (PlayerPrefs.GetInt (levelLocks [i]) == 0) {
                levelUnlock [i] = false;
            } else {
                levelUnlock [i] = true;
            }
            if (levelUnlock [i]) {
                locks [i].SetActive (false);
                levels [i].GetComponent<Button> ().interactable = true;
            }
        }
    }

    // Update is called once per frame
    void Update () {
    }
}
```

Menu.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;
public class Menu : MonoBehaviour {
	void Start(){
	}

	void Update () {
	}

	 void OnMouseDown(){
		SceneManager.LoadScene (SceneManager.GetActiveScene ().buildIndex + 1);
		}
	}
```

Movement.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Movement : MonoBehaviour {
	public List<GameObject> cubosGenerados;
	public List<Vector2> posiciones;
	GameObject timer;
	// Use this for initialization
	void Start () {
		timer = GameObject.Find("Timer");
		Invoke ("GetData", 1);
```

```
	}
	// Update is called once per frame
	void Update () {
		if (timer.activeInHierarchy == false) {
				StartCoroutine (startMove ());
		}
	}

	void GetData(){
		cubosGenerados = GameObject.Find
("ControlLetras").GetComponent<ControllerLetter> ().cubosGenerados;
		posiciones = GameObject.Find ("Positions").GetComponent<LetterControl>
().posiciones;
	}
	IEnumerator startMove(){
		yield return new WaitForSeconds (0.1f);
		for (int i = 0; i < cubosGenerados.Count; i++) {
			if (cubosGenerados [i] != null) {
				cubosGenerados [i].transform.position = Vector2.MoveTowards
(cubosGenerados [i].transform.position, posiciones [i], 10 * Time.deltaTime);
			}
		}
	}
}
```

MovingPlatform.cs

```
using UnityEngine;
using System.Collections;
public class MovingPlatform : MonoBehaviour {
	public float direction;
	public int speed;
	// Use this for initialization
	void Start () {
	}
	// Update is called once per frame
	void Update () {
		transform.Translate (Vector2.right * direction * Time.deltaTime * speed);
		if (this.gameObject.transform.position.x <= -27) {
			transform.position = new Vector2 (40, transform.position.y);
		}
		if (this.gameObject.transform.position.x > 40) {
			transform.position = new Vector2 (-27, transform.position.y);
		}
	}
}
```

Nube.cs

```
using UnityEngine;
using System.Collections;
public class Nube : MonoBehaviour {
	GameObject timer;
	// Use this for initialization
	void Start () {
		timer = GameObject.FindGameObjectWithTag ("Timer");
	}
```

```
        // Update is called once per frame
        void Update () {
                if (timer.activeInHierarchy == false) {
                        transform.Translate (Vector2.right * Time.deltaTime * 5f);
                        if (this.gameObject.transform.position.x > 40) {
                                transform.position = new Vector2 (-27, transform.position.y);
                        }
                }
        }
        }
```

OnExit.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class OnExit : MonoBehaviour {
        SelectClass sC;
        void Start(){
                sC = this.gameObject.GetComponent<SelectClass> ();
        }
        void OnApplicationQuit(){
                Debug.Log ("Test");
                if (SelectClass.tablesCreated == 1) {
                        for (int i = 0; i < 10; i++) {
                                sC.DropLvl ("Lvl" + (i + 1).ToString ());
                        }
                        SelectClass.tablesCreated = 0;
                }
        }
}
```

PlatformSpawn.cs

```
using UnityEngine;
using System.Collections;
public class PlatformSpawn : MonoBehaviour {
	public GameObject platform;
	// Use this for initialization
	void Start () {
	}

	// Update is called once per frame
	void Update () {
		if (platform.activeSelf == false) {
			Invoke ("ReSpawn",1);
		}
	}
	void ReSpawn(){
		platform.SetActive (true);
		platform.transform.position = this.gameObject.transform.position;
		platform.GetComponent<FallPlatform> ().GetComponent<Rigidbody2D>
().isKinematic = true;
		platform.GetComponent<BoxCollider2D> ().isTrigger = false;
	}
}
```

PNGtoTexture2D.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class PNGtoTexture2D  {
	public PNGtoTexture2D(){
	}
	public Texture2D ConvertSprite (Sprite sprite){
		if (sprite.rect.width != sprite.rect.width) {
			Texture2D croppedTexture = new Texture2D ((int)sprite.rect.width,
(int)sprite.rect.height);
			Color[] pixels = sprite.texture.GetPixels ((int)sprite.textureRect.x,
(int)sprite.textureRect.y, (int)sprite.textureRect.width, (int)sprite.textureRect.height);
			croppedTexture.SetPixels (pixels);
			croppedTexture.Apply ();
			return croppedTexture;
		} else {
			return sprite.texture;
		}
	}
}
```

Reload.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;
public class Reload : MonoBehaviour {
	// Use this for initialization
	void Start () {
```

```
	}
	// Update is called once per frame
	void Update () {
	}
	void OnMouseDown(){
		SceneManager.LoadScene("MENU");
	}
}
```

Restart.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;
public class Restart : MonoBehaviour {
	// Use this for initialization
	void Start () {
	}

	// Update is called once per frame
	void Update () {
	}
	public void OnMouseDown(){
		SceneManager.LoadScene (SceneManager.GetActiveScene ().name);
		}
	}
```

SelectClass.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;
using System.Data;
using System.IO;

public class SelectClass : MonoBehaviour {
	dbAccess db;
	private string connectionString;
	string p;
	string[] lvls;
	public static int tablesCreated = 0;
	private string connection;
	// Use this for initialization
	void Start () {
		lvls = new string[10];
		db = GetComponent<dbAccess>();
		for (int i = 0; i < lvls.Length; i++) {
			lvls [i] = "Lvl" + (i + 1).ToString();
		}
	}

	public void CreateNFill(){
		CreateCopy ();
		FillCopy ();
	}
```

```
public void ContentLevels(){
        for (int y = 0; y < lvls.Length; y++) {
                Select10Random (lvls [y]);
                Delete10Random (lvls [y]);
        }
        DropClone ();
        tablesCreated = 1;
}

public void CreateCopy(){
        db.OpenDB ("db_Tesis.sqlite");
        db.CreateCopyDB ();
        db.CloseDB ();
}

public void FillCopy(){
        db.OpenDB ("db_Tesis.sqlite");
        db.FillCopyDB ();
        db.CloseDB ();
}

public void Select10Random(string Lvl){
        db.OpenDB ("db_Tesis.sqlite");
        db.Select10 (Lvl);
        db.CloseDB ();
}
```

```
        public void Delete10Random(string Lvl){
                db.OpenDB ("db_Tesis.sqlite");
                db.Delete10 (Lvl);
                db.CloseDB ();
        }

        public void DropClone(){
                db.OpenDB ("db_Tesis.sqlite");
                db.KillClone();
                db.CloseDB ();
        }

        public void DropLvl(string Lvl){
                db.OpenDB ("db_Tesis.sqlite");
                db.EmptyTableLvl (Lvl);
                db.CloseDB ();
}
}
```

SManager.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;
using System.Collections.Generic;
using UnityEngine.UI;
using System;
public class SManager : MonoBehaviour {
        private CodeTemp ct;
        public Text sman;
        private ContinueVariables contV;
        List<double[]> red;
        List<double[]> resultRedNeu;
        double resultTime;
        double resultLength;
        double resultJump;
        double resultPrevious;
        NiceSceneTransition nct;
        GameObject timer;
        GameObject obj;
        GameObject obj2;
        public int letColect;
        public int index;
        Player plyr;
        public Sprite sprit;
        double desStudy;
        public static double studyorTest = 1;
        AudioClip aud;
```

```
	public List<char> cantLetras;
	public int totalLetters;
	AudioSource source;
	public int minJump;
	public int minTime;
	Clock cl;
	bool result;
	public List<char> answer;
	string word;
	float pitchLvl;
	float doppler;
	int finalTime;
	int finalJump;
	public static int repetitionLvlcount = 0;
	//int trigger = 0;
	// Use this for initialization
	void Start () {
		red = new List<double[]> ();
		resultRedNeu = new List<double[]> ();
		red.Clear ();
		resultRedNeu.Clear ();
		answer = new List<char> ();
		contV = new ContinueVariables ();
		ct = new CodeTemp ();
		nct = GameObject.Find
("NiceSceneTransition").GetComponent<NiceSceneTransition> ();
		cl = GameObject.Find ("OnGoingTimer").GetComponent<Clock> ();
		plyr = GameObject.Find ("Player").GetComponent<Player>();
		source = this.gameObject.GetComponent<AudioSource> ();
```

```
		obj = new GameObject ();
		obj2 = new GameObject ();
		Invoke ("GetData", 0.5f);
		Invoke ("CreateObject", 1f);
	}
	// Update is called once per frame
	void Update () {
	}
	public void TriggerEnd(){
			finalTime = cl.tiempoInt;
			finalJump = plyr.jumpCount;
			obj.SetActive (true);
			obj2.SetActive (true);
			result = CheckMatch (cantLetras, answer);

		if (result == false) {
			GameObject answer = Instantiate (Resources.Load ("Triste",
typeof(GameObject))) as GameObject;
			answer.transform.position = new Vector2 (10, 5);
		} else {
			GameObject answer = Instantiate (Resources.Load ("Feliz",
typeof(GameObject))) as GameObject;
			answer.transform.position = new Vector2 (10, 10);
		}
			StartCoroutine (Calculate ());
			StartCoroutine (EndScene ());
	}
	IEnumerator EndScene(){
		cl.enabled = false;
```

```
            Destroy (plyr.gameObject);
            GameObject boom = Instantiate (Resources.Load ("Particles", typeof(GameObject)))
as GameObject;
            yield return new WaitForSeconds (1);
            Destroy (boom, 1);
            obj.GetComponent<AudioSource> ().Play ();
            yield return new WaitForSeconds(4);
            if (repetitionLvlcount < 10) {
                  nct.LoadScene (SceneManager.GetActiveScene().name);
                  SManager.repetitionLvlcount++;
            } else {
                  PlayerPrefs.SetInt (SceneManager.GetActiveScene().name, 1);
                  SManager.repetitionLvlcount = 0;
                  nct.LoadScene ("StageSelect");
            }
      }
      bool CheckMatch(List<char> l1, List<char> l2){
            if (l1 == null && l2 == null) {
                  return true;
            } else if (l1 == null || l2 == null) {
                  return false;
            }

            if (l1.Count != l2.Count) {
                  return false;
            }

            for (int i = 0; i < l1.Count; i++) {
                  if (l1 [i] != l2 [i])
```

```
                return false;
        }
        return true;
}

void GetData(){
        word = GameObject.Find ("WordGet").GetComponent<WordGet> ().RandomSelect;
        sprit = GameObject.Find ("WordGet").GetComponent<WordGet> ().selectedSprite;
        aud = GameObject.Find ("WordGet").GetComponent<WordGet> ().sound;
        cantLetras = GameObject.Find ("WordGet").GetComponent<WordGet> ().letters;
}
void CreateObject(){
        index = GameObject.Find ("WordGet").GetComponent<WordGet> ().lineIndex;
        //obj = new GameObject ();
        //obj2 = new GameObject ();
        obj.AddComponent<SpriteRenderer> ();
        obj.AddComponent<AudioSource> ();
        obj.GetComponent<SpriteRenderer> ().sprite = sprit;
        obj.GetComponent<AudioSource> ().clip = aud;
        obj.GetComponent<AudioSource> ().playOnAwake = false;
        if (totalLetters >= 6) {
                doppler = 1.2f;
                pitchLvl = 1f;
                obj.GetComponent<AudioSource> ().dopplerLevel = doppler;
                obj.GetComponent<AudioSource> ().pitch = pitchLvl;
        } else {
                doppler = 0.95f;
                pitchLvl = 0.7f;
                obj.GetComponent<AudioSource> ().dopplerLevel = doppler;
```

```
                obj.GetComponent<AudioSource> ().pitch = pitchLvl;
        }
        obj.transform.localScale += new Vector3 (1, 1, 0);
        obj.gameObject.transform.position = this.gameObject.transform.position;
        obj.SetActive (false);
        obj2.AddComponent<MeshRenderer> ();
        obj2.AddComponent<TextMesh> ();
        obj2.GetComponent<TextMesh> ().text = word;
        obj2.GetComponent<TextMesh> ().anchor = TextAnchor.MiddleCenter;
        obj2.GetComponent<TextMesh> ().fontStyle = FontStyle.Bold;
        obj2.GetComponent<TextMesh> ().color = Color.black;
        obj2.SetActive (false);
        obj2.transform.localScale += new Vector3 (1, 1, 0);
        obj2.gameObject.transform.position = new Vector3 (0, 4, 0);
}

public void playAudio(){
        source.pitch = pitchLvl;
        source.dopplerLevel = doppler;
        source.clip = aud;
        source.Play ();
}

IEnumerator Calculate(){
        yield return new WaitForSeconds (0);
        double[] a = new double[4];
        resultTime = contV.GetTime (finalTime, minTime);
        a [0] = resultTime;
        resultLength = contV.GetPalabrasCorrectas (cantLetras, answer);
```

```
                a [1] = resultLength;
                resultJump = contV.GetJumps (finalJump, minJump);
                a [2] = resultJump;
                resultPrevious = contV.previousLvl (studyorTest);
                a [3] = resultPrevious;
                red.Add (a);
                resultRedNeu = ct.red_neuronalTrue ("w1", "w2", red);
                foreach (double[] d in resultRedNeu) {
                        desStudy = d[0];
                }
                sman.text = desStudy.ToString ();
                if (desStudy < 0.5f) {
                        studyorTest = -1;
                } else {
                        studyorTest = 1;
                }
}
}
        // Use this for initialization
```

TestNew.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class TestNew : MonoBehaviour {
        public List<double[]>  red;
        CodeTemp li = new CodeTemp();
        // Use this for initialization
```

```
        void Awake(){
                red = new List<double[]> ();
                red = li.red_neuronal ("w1", "w2", "data");
        }
        void Start () {
                li.show_matrix (red);
                foreach (double[] test in red) {
                        Debug.Log (test[0]);
                }
        }
        // Update is called once per frame
        void Update () {
        }
}
```

TextClass.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class TextClass : MonoBehaviour {
        public Text txt;
        // Use this for initialization
        void Start () {
        }
        // Update is called once per frame
        void Update () {
                txt.text = "Clase " + ButtonControl.option + " escogida";
        }
}
```

TimerController.cs

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;
using System.Collections.Generic;
public class TimerController : MonoBehaviour {
	List<char> size;
	public Text timer;
	public float countdown;
	public float countdownNormal;
	float tiempo = 0;
	int tiempoInt;
	// Use this for initialization
	void Start () {
		size = GameObject.Find ("WordGet").GetComponent<WordGet> ().letters;
		if (size.Capacity <= 6) {
			countdown = 10;
		} else {
			countdown = 15;
		}
		countdownNormal = countdown;
		timer.text = ":";
		timer.color = Color.white;
		timer.fontSize = 15;
		timer.fontStyle = FontStyle.Bold;
	}
	// Update is called once per frame
	void Update () {
		size = GameObject.Find ("WordGet").GetComponent<WordGet> ().letters;
```

```
                tiempo += Time.deltaTime;
                tiempoInt = (int) tiempo;
                if (countdownNormal > 0) {
                        countdownNormal = countdown - tiempoInt;
                }
                timer.text = ": " + countdownNormal.ToString();
                if (size.Capacity < 6) {
                        if (tiempoInt >= 7) {
                                timer.color = Color.red;
                        }
                }
                else
                        if (tiempoInt >= 12) {
                                timer.color = Color.red;
                        }
                if(countdownNormal == 0){
                        gameObject.SetActive (false);
                }
        }
}
```

UIManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class UIManager : MonoBehaviour {
```

```
GameObject[] pauseObjects;
bool pauseOn = false;
// Use this for initialization
void Start () {
	Cursor.visible = true;
	Time.timeScale = 1;
	pauseObjects = GameObject.FindGameObjectsWithTag("ShowOnPause");
	hidePaused();
}

// Update is called once per frame
void Update () {
	if (pauseOn == true) {
		if (Time.timeScale == 1) {
			Time.timeScale = 0;
			showPaused ();
			pauseOn = false;
		} else if (Time.timeScale == 0) {
			Time.timeScale = 1;
			hidePaused ();
			pauseOn = false;
		}
	}
}
public void hidePaused(){
	foreach(GameObject g in pauseObjects){
		g.SetActive(false);
	}
}
```

```
	public void showPaused(){
		foreach(GameObject g in pauseObjects){
			g.SetActive(true);
		}
	}

	public void pauseControl(){
		if(Time.timeScale == 1)
		{
			Time.timeScale = 0;
			showPaused();
		} else if (Time.timeScale == 0){
			Time.timeScale = 1;
			hidePaused();
		}
	}

	public void Pause(){
		pauseOn = true;
	}
}
```

Wind.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Wind : MonoBehaviour {
	public float windForce;
```

```
        public int X;
        public int Y;
        // Use this for initialization
        void Start () {
        }
        // Update is called once per frame
        void Update () {
        }

        void OnTriggerStay2D(Collider2D obj){
                if (obj.gameObject.CompareTag ("Player")) {
                        obj.GetComponent<Player> ().rb.AddForce (new Vector2(X,Y) * windForce);
                }
        }
}

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.IO;
```

WordGet.cs

```
public class WordGet : MonoBehaviour {
        public List<DataDB> dataDB;
        public List<string> selectedWordENG;
        public List<string> selectedWordSPA;
        [HideInInspector]
        public List<string> selectedWordIMG;
```

```
ImageConvertion imgc;
[HideInInspector]
public List<string> selectedSOUND;
public List<string> selectedANS;
public int lineIndex;
public Sprite selectedSprite;
[HideInInspector]
public string audiostring;
public string RandomSelect;
public string RandomAns;
char[] eachletter;
public List<char> letters;
public int lettercount;
public AudioClip sound;
// Use this for initialization
void Start () {
        imgc = new ImageConvertion ();
        dataDB = GameObject.Find ("AccesDB").GetComponent<DBAccess1> ().dataDB;
        Invoke("GetDB",0.1f);
        StartCoroutine(SelectRandom());
}
// Update is called once per frame
void Update () {
}



private void GetDB(){
```

```
            for (int i = 0; i < dataDB.Count; i++) {
                DataDB db = dataDB [i];
                selectedWordENG.Add (db.ENG);
                selectedWordSPA.Add (db.SPA);
                selectedSOUND.Add (db.SOUND);
                selectedWordIMG.Add (db.IMG);
                selectedANS.Add (db.ANS);
            }
        }

        IEnumerator SelectRandom(){
            yield return new WaitForSeconds (0.1f);
            int random = Random.Range(0,selectedWordENG.Count);

            RandomSelect = selectedWordENG [random];
            lineIndex = selectedWordENG.IndexOf (RandomSelect);
            eachletter = RandomSelect.ToCharArray ();
            letters.AddRange (eachletter);
            lettercount = letters.Count;
            RandomAns = selectedANS[lineIndex];
            selectedSprite = imgc.Base64ToImage (selectedWordIMG [lineIndex]);
            audiostring = selectedSOUND [lineIndex];
            sound = imgc.Base64ToAudio (audiostring, RandomSelect);
        }
}
```