23. Tecnologías de la Información y la Comunicación

Búsqueda de imágenes basada en el Procesamiento de Lenguaje Natural aplicada en una Agenda Visual orientada a Personas con Trastorno del Espectro Autista

Zorrilla Villanueva, Andrés; andreszorrillav@gmail.com
Facultad de Ingeniería
Universidad Nacional de Itapúa

Resumen

Se desarrolló una aplicación de agenda visual para Personas con Trastorno del Espectro Autista aplicando técnicas de *Procesamiento de Lenguaje Natural* (PLN) considerando la búsqueda textual de las imágenes por su descripción de contenido, buscando obtener resultados de acuerdo al contenido semántico de las descripciones asociadas a la imagen, a diferencia de las aplicaciones de este tipo en el mercado. Para la implementación del PLN, se ha entrenado un conjunto de documentos utilizando un algoritmo denominado *Skip-gram,* que consiste en la extracción de texto en porciones donde se toma una *palabra objetivo* con las *palabras de su contexto,* y se va entrenando en una red neuronal. La búsqueda consistió en la vectorización de las descripciones como también el término de búsqueda ingresado. Se obtuvieron los resultados en un orden de relevancia de acuerdo a la similaridad coseno entre el término ingresado y las descripciones de la base de datos. Para probar la búsqueda implementada, se han realizado comparaciones entre la similaridad coseno entre vectores extraídos del modelo de *Skip-gram,* y la comparación de caracteres entre término de consulta y descripciones. Se pudo observar mayor precisión en la búsqueda basada en el algoritmo de *Skip-gram.* Finalmente se ha desarrollado una aplicación con el framework Django permitiendo la *Gestión de Agendas Visuales* integrando la búsqueda propuesta.

Palabras Clave: trastorno del espectro autista, procesamiento de lenguaje natural, word embeddings, skip-gram, agenda visual.

Introducción

El Trastorno del Espectro Autista (TEA) se define como "(...) una disarmonía generalizada en el desarrollo de las funciones cognitivas superiores e independiente del potencial intelectual inicial" (Quijada, 2008). Las personas con TEA presentan principalmente dificultades con la interacción social, problemas con expresión del lenguaje, comunicación y el pensamiento.

Uno de los recursos de comunicación aplicados para la estimulación en la comunicación son las *Agendas Visuales*, que son una secuencia de imágenes que representan las actividades diarias y a menudo resúmenes simples de sucesos relevantes del día (Cabeza Pereiro, 2018).

Actualmente existen aplicaciones que permiten la gestión dinámica de agendas, tales como *LetMeTalk*[1] y *TeApps*[2]. Para la confección de agendas visuales, estas herramientas permiten la obtención de imágenes mediante comparación de etiquetas de texto predefinidas asociadas a la imagen y el texto de consulta. Sin embargo, solo se obtienen resultados que coinciden con las etiquetas de la imagen, sin tener el cuenta el significado semántico del texto de consulta. Esto puede obtener resultados no esperados, o con menor precisión a lo que se desea obtener.

Teniendo en cuenta lo expuesto, en esta investigación se propone desarrollar una agenda visual basada en la descripción de contenidos de las imágenes utilizando Procesamiento de Lenguaje Natural (PLN) para mejorar el criterio de búsqueda a las aplicaciones existentes actualmente, ya que se tendrá en cuenta la similaridad semántica durante el proceso de búsqueda, es decir, aquellas descripciones que tienen un significado, sentido o interpretación igual. Para la implementación de PLN se tiene en cuenta distintas áreas de la ciencia computacional, tales como: la inteligencia artificial, el aprendizaje automático o la inferencia estadística; con la lingüística (Abalos Serrano, 2016).

Objetivos

Objetivo General

Desarrollar una aplicación de agenda visual aplicando técnicas de procesamiento de lenguaje natural considerando la búsqueda de las imágenes por su descripción de contenido.

Objetivos Específicos

- Analizar y definir los requerimientos de la aplicación con profesionales en el área.
- Implementar la búsqueda propuesta utilizando librerías existentes y recolectando conjunto de imágenes y archivos de texto.
- Implementar la agenda visual

[1] *LetMeTalk*: vease en: http://www.letmetalk.info/es
[2] *Teapps*: vease en: http://aulautista.wixsite.com/teapps

integrando con la base de conocimiento generada.
- Validar los resultados obtenidos de las búsquedas.

Materiales y Métodos

Definición de herramienta: Agenda Visual

Hernáez Cuesta & García Bedia (n.d.) afirma que las *agendas visuales*, son herramientas cuya función es suministrar información visual en forma lógica, estructurada y secuencial. Son utilizadas por profesionales en la terapia cognitivo conductual, un tratamiento efectuado a los pacientes que se focaliza en modificar los comportamientos y pensamientos que gestionan un problema psicológico que se quiere intervenir (Gratacós, n.d.).

A través de actividades (expresadas por la agenda) se busca que el paciente sea capaz de adaptarse al entorno, modular la conducta, y enseñarle habilidades.

Entre las imágenes utilizadas están los sistemas de pictogramas, es decir, recursos gráficos que se caracterizan por su facilidad de interpretación dado a los iconos que representan de forma clara el concepto que desean transmitir.

Requerimientos de la Aplicación

Se realizó una aplicación que permita la gestión de actividades de una agenda visual, a través de secuencia de imágenes. Cada imagen cuenta con un estado, para establecer un estímulo o refuerzo en lo que se quiera hacer aprender al paciente.

Para la obtención de las imágenes se implementó búsquedas por texto utilizando Procesamiento de Lenguaje Natural, y utilizando una técnica denominada *Word Embeddings,* que es el entrenamiento de redes neuronales artificiales basado en documentos no anotados para la obtención de una base de conocimiento que permite la representación de palabras y su posterior evaluación con el término de búsqueda.

Este método presenta ventajas ya que se tiene en cuenta la información adicional como la semántica, la estructura, la secuencia y el contexto de las palabras cercanas en base a los documentos sin procesar (Torres López & Arco García, 2016).

Metodología Utilizada

La metodología utilizada es *Scrum*, la cual consiste en un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto (Proyectos Ágiles, 2017).

Para esta investigación se realizaron *sprints* semanales y quincenales dependiendo de la tarea propuesta, donde cada tarea tiene una duración de uno a dos días.

Herramientas Utilizadas

Para lograr el objetivo propuesto se utilizaron las siguientes herramientas:

1. Desarrollo: Python, Cython, Gensim, Django, MySQL, NGinx.
2. Colaborativas: GanttProject, Git, SmartGit, StarUML.

Arquitectura Física

En base al entorno de programación elegido, se optó por la arquitectura cliente-servidor donde la aplicación y su base de datos se encuentra alojada en un servidor, donde los usuarios acceden a la aplicación desde un navegador web.

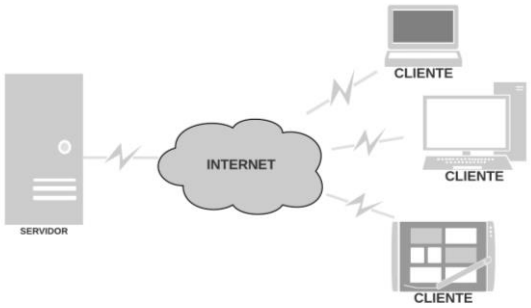


Figura 1: Arquitectura Física.

Arquitectura Lógica

Se utiliza la arquitectura MVT (*Model*, View, *Template*) que utiliza el *framework Django.* Además de la arquitectura MVT, ver Figura 2, la aplicación contiene procesos externos, que no interactúan directamente con el usuario, pero son esenciales para el funcionamiento del sistema. tales como la carga de imágenes, los procesos de traducción y limpieza de datos, los algoritmos de entrenamiento, reentrenamiento y de conversión de descripciones de imágenes a vectores.

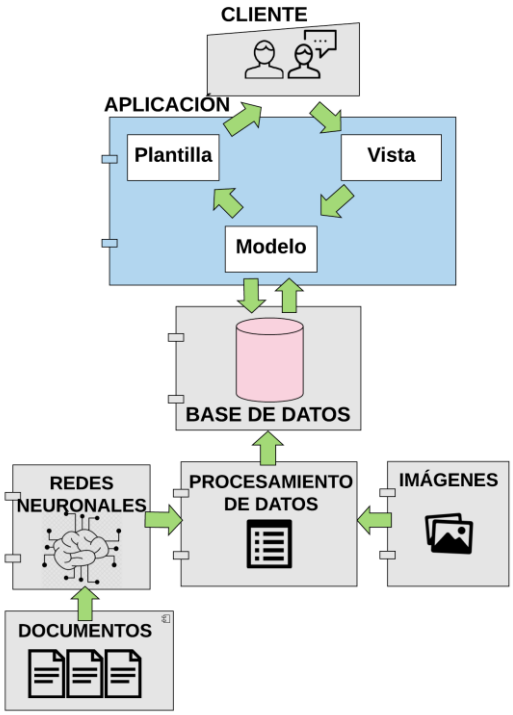


Figura 2: Arquitectura Lógica.

Búsqueda de Imágenes

Cuando se realiza una búsqueda se requiere un proceso de representación, almacenamiento y organización de la información para conseguir el acceso y recuperación de información que responda a las necesidades de un usuario (Epifanio Tula & Medeot, 2007).

Este proceso de la información se ha realizado mediante PLN, utilizando cantidades de información en formato texto con un grado de eficacia aceptable mediante técnicas de representación de palabras en un espacio vectorial a través de un modelo denominado *Word Embeddings.*

Word Embeddings

Brownlee (2017) explica que *Word Embeddings* consiste en un conjunto de lenguajes de modelado y técnicas de aprendizaje donde las palabras que tienen un mismo significado semántico tienen una representación similar. En la Figura 3 se observa que los vectores están agrupados según una característica en común.

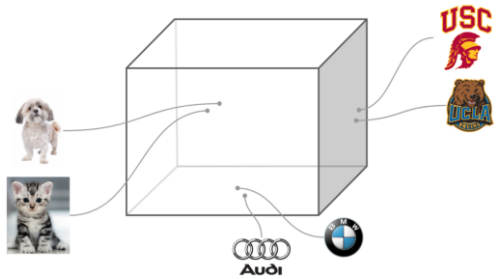


Figura 3: Representación de Palabras en un Espacio Vectorial (Mishra, 2017).

Una palabra se representa por un vector de valor real de una determinada dimensión, promediado con vectores palabras en un contexto.

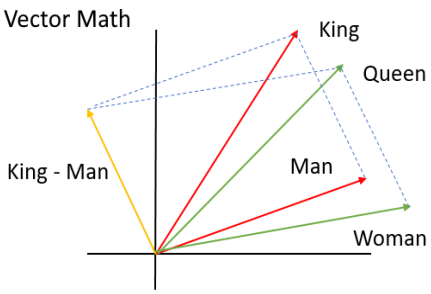


Figura 4: Analogía entre palabras (Shure, 2017).

En la figura 4 puede verse como se exhibe una propiedad de Word Embeddings: las *analogías entre palabras*, donde relaciones lineales entre pares de palabras son indicativas de la calidad de la similaridad para que pueda establecerse una métrica de comparación.

Algoritmo de Skip-gram.

El objetivo de entrenamiento del modelo *Skip-gram* es encontrar representaciones de palabras que sean útiles para predecir las palabras que rodean en una oración o un documento (Mikolov, Sutskever, Chen, Corrado & Dean, 2013). Se extrae una palabra específica del medio de una oración, y también toma una a una las palabras que lo rodean dentro de una *ventana* definida para luego alimentar una *red neuronal*. Esta ventana es el rango de palabras cercanas según una palabra específica dentro de una oración. Esto se puede ver en la Figura 5.

Figura 5: Representación de una palabra dentro de un contexto (Siriwardhana, 2017).

Para la implementación se utiliza una red neuronal con una capa oculta. Las palabras tienen una representación atómica de *vectores one-hot,* que consisten vector

binario que es todos los valores cero excepto el índice del entero, que está marcado con un 1 (Brownlee, 2018).

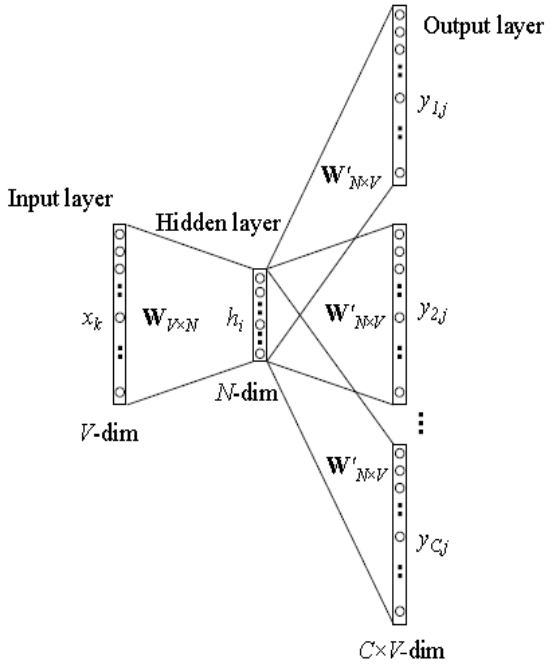


Figura 6: Modelo Skip-gram (Roung, 2016).

En la Figura 6 se puede observar la arquitectura del modelo *Skip-gram*, dado una entrada **X** y salidas del contexto **Y**.

Minnaar (2015) explica que:

- **X** representa el vector *one-hot* palabra de entrada.
- {$\mathbf{y_1}$, ..., $\mathbf{y_c}$} son los vectores *one-hot* codificados a las palabras de salida.
- La matriz $\boldsymbol{W_{V\times N}}$ es la matriz de pesos entre la capa de entrada y la capa oculta cuya *i-ésima* fila representa los pesos correspondientes a la palabra en el vocabulario.
- Cada vector de salida también tiene una producto $N \times V$ asociada como matriz **W'**.

Para la propagación hacia adelante, se establece la entrada en una capa oculta hi es simplemente la suma de los pesos de sus entradas.

$$h = x_T \mathbf{W} = W_{k,\cdot} = v_{wi}$$

Esta matriz de pesos **W** contiene los vectores codificados de todas las palabras del vocabulario (Minnaar, 2015).

Además Minnaar (2015) expresa que en las capas de salida, para cada palabra de salida comparten los mismos pesos, por lo tanto, $u_{c,j} = u_j$, Finalmente, se puede calcular el resultado del *j-ésimo* nodo de la palabra *c-ésima* salida través de la *función softmax*[3]

$$p(w_{c,j} = w_{O,c}|w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{j'})}$$

Este valor es la probabilidad de que la salida del *j-ésimo* nodo de la palabra de salida sea igual al valor real del vector de salida c-ésima (que es uno vector one-hot).

Las Redes Neuronales se basan en el aprendizaje de los pesos en una regla de ajustes de error, de manera que las salidas de la red coincidan con las salidas

[3] Una función softmax consiste en distribución de probabilidad sobre un conjunto de etiquetas mutuamente excluyentes.

deseadas, o por lo menos las que sean posibles (Palma Méndez & Marín Morales, 2008, p.650-651), por lo que se establece los gradientes de error necesarios para el algoritmo de retropropagación para aprender tanto **W**:

$$w_{ij}^{\prime(new)} = w_{ij}^{\prime(old)} - \eta \cdot \sum_{c=1}^{C} (y_{c,j} - t_{c,j}) \cdot h_i$$

Como para **W'**:

$$w_{ij}^{(new)} = w_{ij}^{(old)} - \eta \cdot \sum_{j=1}^{V} \sum_{c=1}^{C} (y_{c,j} - t_{c,j}) \cdot w'_{ij} \cdot x_j$$

Obtención de Corpus

El corpus basado en el lenguaje español, que consiste en de casi 1.500 millones de palabras, compilado a partir de diferentes recursos de la web[4] .

Se utilizó la libreria *gensim* de *Python* para el procesamiento de texto para la generación del corpus.

Cada palabra está compuesta de 300 *características*.

Imágenes con Descripciones de Contenido.

Las imágenes obtenidas consisten en datos, con etiquetas simples, sus sinónimos y una definición. En total se obtuvieron 15.395 pictogramas a color[5].

[4] Se ha obtenido el corpus de: Cristian Cardellino. Spanish Billion Words Corpus and Embeddings (Marzo 2016), http://crscardellino.me/SBWCE/

[5] Se ha obtenido las imágenes de: Sergio Palao ARASAAC (http://arasaac.org) Licencia: CC (BY-NC-SA) Propiedad: Gobierno de Aragón.

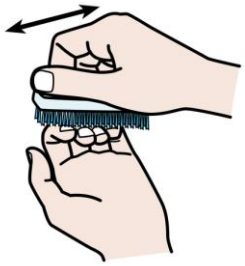
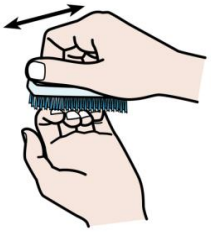


Figura 7: Ejemplo de un pictograma (Palao, 2013).

Procesamiento de Datos

Una vez obtenidas las imágenes con sus respectivas descripciones se ha procedido a establecer un formato en un archivo de texto para la carga de archivos a la aplicación, luego de un proceso de traducción, corrección, limpieza y formato.



```
cepillo.png

-Limpiar manos
-Cepillando las uñas
-Dos manos limpiándose
con un cepillo
```

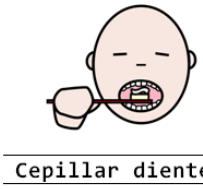
```
morder_3.png

-niño mordiendo brazo
-niño blanco muerde brazo
-un niño enojado muerde
a una persona
```

Figura 8: Ejemplo del procesamiento de imágenes con sus descripciones (Palao, 2013).

Carga de Datos

Al cargar las imágenes en la base de datos, se cargan las descripciones y su vector de palabras del corpus.

Debido a que las descripciones son composiciones de una o más palabras, se estableció el *promedio de vectores*, donde se halla el promedio de las palabras de la descripción, obteniendo de esta forma un vector resultante de la misma dimensión. Estos vectores se codifican a texto y se guardan en la base de datos para evitar realizar este cálculo del promedio de vectores en cada iteración.



Cepillar dientes

$$\mathbf{e}_{\text{cepillar}} = \begin{bmatrix} -1 \\ 0.51 \\ 0.93 \\ -0.09 \end{bmatrix} \quad \mathbf{e}_{\text{dientes}} = \begin{bmatrix} 1 \\ 0.12 \\ -0.62 \\ 0.51 \end{bmatrix}$$

$$\mathbf{e}_{\text{pr(cepillar+dientes)}} = \begin{bmatrix} 0 \\ 0.315 \\ 0.155 \\ 0.21 \end{bmatrix}$$

Figura 9: Vectorización de una descripción asociada a una imagen (Palao, 2013).

Métrica de Similaridad

Perone (2013) dice que la similitud del coseno entre dos vectores en un espacio vectorial es una medida que calcula el coseno del ángulo entre ellos.

De la fórmula de producto escalar de vectores puede obtenerse la similaridad coseno, basado en la siguiente ecuación:

$$\vec{a} \cdot \vec{b} = \|\vec{a}\|\|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|}$$

Gallo (2016) afirma que si los vectores son casi paralelos, se supone que ambas oraciones son similares. Mientras que si los vectores son ortogonales, entonces se supone que las oraciones son independientes o no similares. En la Figura 10 se expresa un ejemplo en un espacio vectorial donde los vectores son próximos.

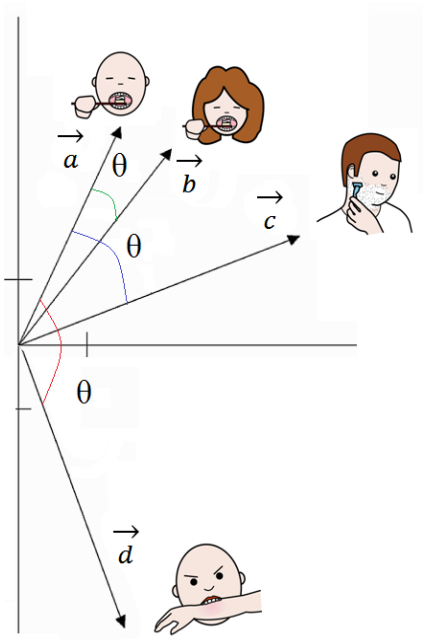


Figura 10: Representación de descripciones de imágenes en un espacio vectorial demostrando la similitud entre las descripciones más cercanas semánticamente (Palao, 2013).

Implementación de la búsqueda

Al realizar una búsqueda, se introduce un texto de consulta, el cual contiene la información que se quiere recuperar.

Este texto de consulta se expresa como vector con el promedio de vectores.

Luego se va comparando el vector de consulta con los vectores de la base de datos, y se van guardando los cálculos similaridad del coseno en una cola de prioridad, ordenados de mayor a menor, dejando a los vectores con mejor resultado en primer lugar. Todo este proceso puede verse en la Figura 11.

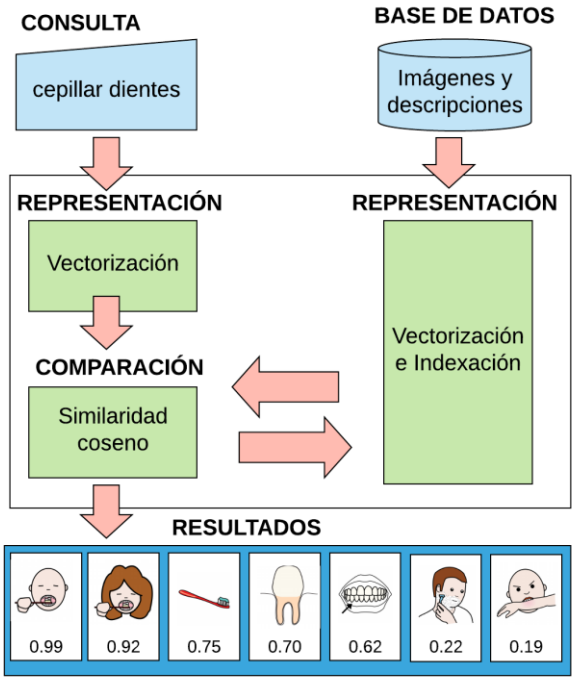


Figura 11: Proceso de recuperación de imágenes según texto de consulta (Palao, 2013).

Resultados y Discusión

Comparación entre búsqueda textual y búsqueda utilizando PLN

Para determinar el grado de precisión de la búsqueda propuesta, se han realizado un conjunto de pruebas entre comparaciones léxicas de descripciones de consulta y base de datos, y utilizando el modelo generado por el algoritmo *Skip-gram* codificando descripciones con el promedio de vectores y ordenando los resultados de acuerdo a la similaridad coseno obtenida.

Para cada caso se clasificaron descripciones en la base de datos y datos de consulta. Se ha considerado como parámetro de precisión la *cantidad de resultados devueltos* (debido a que los resultados esperados se encuentran en los primeros lugares) y el *porcentaje de similitud* que se calcula por medio de porcentaje de aciertos sobre la cantidad de muestras realizadas.

Se han clasificado las descripciones de acuerdo a su cantidad de palabras que quedan de la siguiente manera:

Resultado 1: Descripciones con 1 a 4 palabras.

Con una muestra de 1000 elementos en base de datos y 3957 descripciones de consulta se obtuvieron los siguientes resultados:

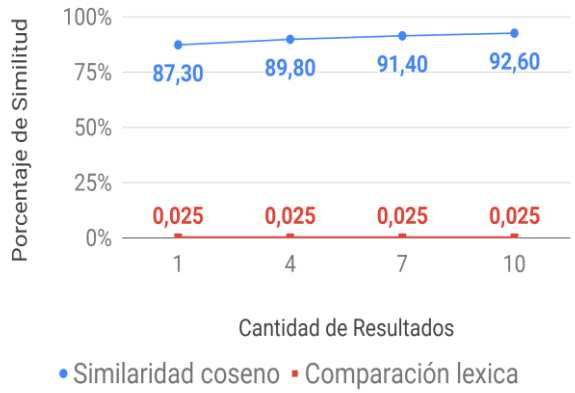


Figura 12: Resultados para descripciones con 1 a 4 palabras.

Se puede observar que existe mayor aproximación exactitud con la métrica de similaridad coseno.

Resultado 2: Descripciones de longitud 5 a 10 palabras.

Con una muestra de 212 elementos en base de datos y 347 descripciones de consulta se obtuvieron los siguientes resultados:

Figura 13: Resultados para descripciones con 5 a 10 palabras

Se puede observar que las palabras todavía dan un resultado óptimo pero va bajando la precisión de la técnica propuesta.

Resultado 3: Descripciones de longitud 10 a 22 palabras.

Con una muestra de 1000 elementos en base de datos y 4000 descripciones de consulta se obtuvieron los siguientes resultados:

Figura 14: Resultados para descripciones con 10 a 22 palabras.

Funcionalidades de la aplicación “Oikoitea”

El sistema desarrollado se realizó en *framework* web *Django*.

Se contemplaron las siguientes funcionalidades para el desarrollo del sistema de gestión de Agendas Visuales:

Requisitos no funcionales

- Autenticación de cuentas de usuario.
- Perfil de usuario para profesionales.
- Gestión de personas con TEA.
- Perfil de usuario para profesionales.
- Gestión de personas con TEA.

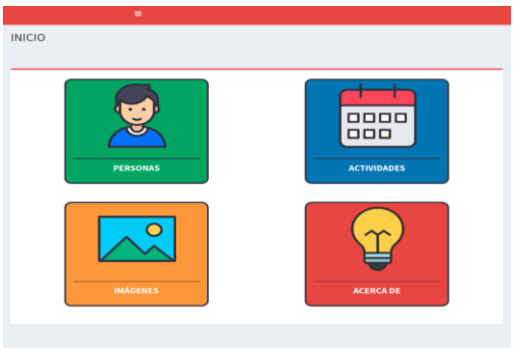


Figura 15: Pantalla de Inicio.

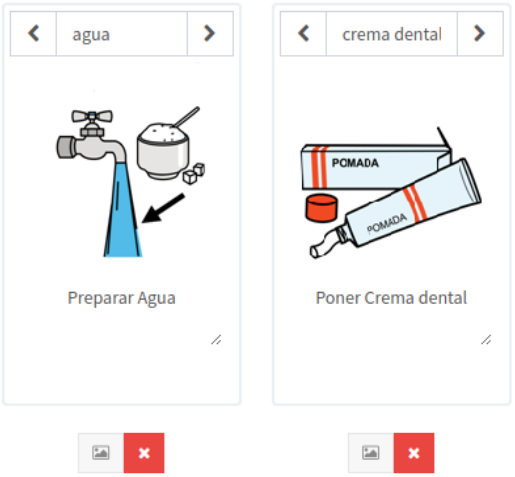


Figura 16: Confección de actividades de una agenda visual mediante búsquedas con PLN.

Requisitos Funcionales

- Gestión de Agendas Visuales: estas permiten la gestión de rutinas y acciones a través de secuencia de actividades que elabora el profesional destinada a personas con TEA.
- Búsqueda de imágenes según su descripción de contenido: A través de esta funcionalidad se busca obtener imágenes precisas según el término de búsqueda introducido, utilizando técnicas de procesamiento de lenguaje natural.
- Cada actividad consta de una imagen y un campo de texto que describe la actividad.
- Además, las actividades cuentan con un estado el cual ayuda a determinar el progreso en la agenda visual, mostrando si la actividad está en proceso, si se realizó de manera correcta, o si no se logró.

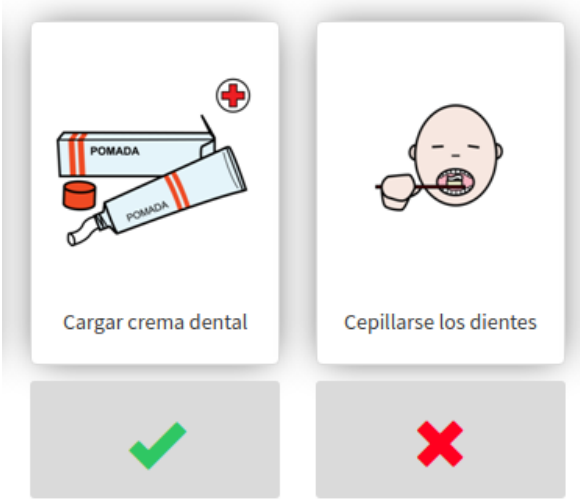


Figura 17: Estados de una actividad.

Conclusiones

El presente trabajo tuvo como objetivo brindar una herramienta que permita la gestión de agendas visuales utilizando como mejora la implementación de técnicas y modelos de PLN para obtener mejores resultados a la hora de realizar una búsqueda textual. Se demostró que la propuesta de *Word Embeddings* presenta diferencias considerables frente a los distintos mecanismos utilizados por las aplicaciones existentes, mostrando mejores resultados ya que se tiene en cuenta la estructura semántica del texto del término de búsqueda.

A medida de que se incrementa la longitud de las descripciones, disminuye la precisión. Por lo tanto se propone como trabajos futuros, la implementación de métodos de alineación entre descripciones y la automatización en el reentrenamiento de la base de conocimiento, agregando los datos de las nuevas descripciones a la base de datos de modo a que se obtenga una base de conocimiento ideal.

Bibliografía

Abalos Serrano, N. (2016). *Aplicaciones del Procesamiento del Lenguaje Natural.* Recuperado de https://www.beeva.com/beeva-view/innovacion/aplicaciones-del-procesamiento-de-lenguaje-natural/

Cabeza Pereiro, E. (2018). *Qué son las agendas visuales para los niños con autismo.* Recuperado de https://www.guiainfantil.com/articulos/educacion/aprendizaje/que-son-las-agendas-visuales-para-los-ninos-con-autismo/

Gratacós, M. (n.d.) *Terapia Cognitivo Conductual: Características y 5 Técnicas.* Recuperado de https://www.lifeder.com/terapia-cognitivo-conductual/

Hernáez Cuesta, C., & García Bedia, A. (n.d.). *Agendas Visuales para la Organización Escolar.* Recuperado https://www.educantabria.es/docs/centros/atencion_a_la_diversidad/CREE/agendas_visuales.pdf

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality.* CoRR, abs/1310.4546.

Mishra, P. (2017). *Word Embeddings with Gensim – Becoming Human: Artificial Intelligence Magazine.* Recuperado de https://becominghuman.ai/word-embeddings-with-gensim-68e6322afdca

Minnaar, A. (2015). *Word2Vec Tutorial Part I: The Skip-Gram Model* [Ebook] (pp. 1-6). Alex Minnaar. Recuperado de http://mccormickml.com/assets/word2vec/Alex_Minnaar_Word2Vec_Tuto

rial_Part_I_The_Skip-Gram_Model.pdf

Perone, C. (2013). *Machine Learning :: Cosine Similarity for Vector Space Models (Part III) | Terra Incognita.* Recuperado de http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/

Palao, S. (2013). *Arasaac. Portal Aragonés de la Comunicación Aumentativa y Alternativa.* Recuperado de: http://www.arasaac.org/

Palma Méndez, J., & Marín Morales, R. (2008). *Inteligencia artificial* (pp. 649-653). Aravaca, Madrid: McGraw-Hill/Interamericana de España.

Quijada, C. (2008). *Espectro Autista. Santiago: Scientific Electronic Library Online SciELO Chile.* Recuperado de: http://www.scielo.cl/pdf/rcp/v79s1/art13.pdf

Shure, L. (2017). *Math with Words – Word Embeddings with MATLAB and Text Analytics Toolbox.* Recuperado de: https://blogs.mathworks.com/loren/2017/09/21/math-with-words-word-embeddings-with-matlab-and-text-analytics-toolbox/

Torres López, C., & Arco García, L. (2016). *Representación textual en espacios vectoriales semánticos. Revista Cubana De Ciencias Informáticas, 10(2), 148 - 180*. Recuperado de https://rcci.uci.cu/?journal=rcci&page=article&op=view&path%5B%5D=1236