Universidad Zaragoza

Trabajo Fin de Máster

SISTEMA DE DIÁLOGO PARA LA COMUNICACIÓN AUMENTATIVA Y ALTERNATIVA CON PICTOGRAMAS

DIALOGUE SYSTEM FOR ALTERNATIVE AND AUGMENTATIVE COMMUNICATION WITH PICTOGRAMS

Autor

Miguel Abadía Gadea

Director

Eduardo Lleida Solano

Escuela de Ingeniería y Arquitectura
Instituto de Investigación en Ingeniería de Aragón

Diciembre 2016

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. MIGUEL ABADÍA GADEA,

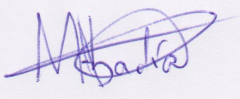
con nº de DNI 73029795B en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) MÁSTER, (Título del Trabajo)

SISTEMA DE DIÁLOGO PARA LA COMUNICACIÓN AUMENTATIVA Y ALTERNATIVA CON PICTOGRAMAS.

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 22 DE NOVIEMBRE DE 2016



Fdo: MIGUEL ABADÍA GADEA

Agradecimientos

En primer lugar, dar las gracias a Eduardo, el referente en este camino, por su paciencia y confianza, fundamentales para desarrollar este proyecto.

Al grupo ViVoLab, en especial a Victoria, Julia, Jorge, Adolfo, Nacho, Pablo y Diego, por permitirme trabajar en un ambiente envidiable.

A mis amigos, por estar ahí siempre, en especial a Adrián, Ángel y Víctor. A María por su apoyo constante.

Finalmente, agradecer a mi familia, de la que es un orgullo formar parte, cuyo aliento es vital cada día.

Sistema de diálogo para la comunicación aumentativa y alternativa con pictogramas.

RESUMEN

En primer lugar, destacar que este Trabajo Fin de Máster se ha desarrollado dentro del grupo de investigación ViVoLab (Voice Input Voice Output Laboratory) de la Universidad de Zaragoza.

Este proyecto se enmarca dentro del campo de la comunicación aumentativa y alternativa, la cual hace más sencilla la relación de personas con necesidades especiales, y tiene como objetivo la creación de un sistema de diálogo. Este sistema de diálogo permite la comunicación mediante texto y pictogramas, e incluye una herramienta de predicción que facilita su uso.

El sistema de diálogo se basa en una aplicación tipo chat que permite a los usuarios enviar mensajes al resto de personas conectadas. Los mensajes son enviados usando tanto texto como pictogramas, y pueden ser introducidos de las dos formas, bien usando un teclado para escribir texto o seleccionando los pictogramas de la propia aplicación. Estos pictogramas están ordenados por categorías gracias al uso de la base de datos de Arasaac (portal Aragonés de la Comunicación Aumentativa y Alternativa).

El sistema de predicción ordena las categorías presentando al usuario las que tienen un uso más probable en cada momento. Dentro de cada categoría, se pueden seleccionar los pictogramas (los cuales también aparecen ordenados por probabilidad de uso) añadiendo una nueva palabra al mensaje que se está escribiendo. El algoritmo de predicción también permite al usuario escoger sus frases más comunes al comienzo de la conversación, así como cada vez que envía o recibe un mensaje para poder seguir la conversación con más fluidez. Si el usuario se decide por redactar una nueva frase, el sistema de predicción le ayudará para completarla, actualizándose cada vez que se introduzca una nueva palabra y/o pictograma.

En cuanto al sistema de predicción, éste se basa en modelos de lenguaje de N-gramas. Los N-gramas son secuencias de palabras de longitud N y la predicción se realiza a través de la probabilidad de cada N-grama en el modelo de lenguaje. Además, el algoritmo utiliza un modelo de lenguaje global y otro modelo de lenguaje caché, basado en el uso qué cada usuario hace del sistema, personalizándolo.

Finalmente, se observan que las prestaciones del sistema mejoran gracias a la caché, en base a los resultados obtenidos. Cuando únicamente el modelo de lenguaje global es utilizado en el algoritmo, un 9.15 % de las palabras son predichas correctamente y se introducen 1.10 palabras por pulsación (es decir, cada vez que un usuario del sistema del diálogo utilizara la predicción sugerida, añadiría 1.10 palabras a la frase que está escribiendo). Además, se consigue un 8.37 % de precisión y un 14.85 % de recall. Estos resultados demuestran la baja calidad del modelo de lenguaje global.

En cambio, cuando se usa únicamente la caché se obtienen unos mejores resultados. Se ha alcanzado un 84.42 % de palabras acertadas por el algoritmo predictor, 3.57 palabras por pulsación, 61.15 % de precisión y 67.91 % de recall.

Dialogue system for alternative and augmentative communication with pictograms.

ABSTRACT

First of all, I would like to stand out that this Master Thesis has been developed in ViVo-Lab (Voice Input Voice Output Laboratory), research group which belongs to *Universidad de Zaragoza*.

This project falls into the framework of the augmentative and alternative communication, which makes easier the relationship of people with special needs. This dialogue system allows the communication through the use of text and pictograms, and it includes a prediction tool which facilitates its usage.

The dialogue system is based on a chat application which permits to the users send messages to the rest of the people connected. The messages are sended using both text and pictograms, and they can be entered by two ways; using a keyboard to write text or choosing pictograms from the app itself. These pictograms are categorized, thanks to the Arasaac's database (Augmentative and Alternative Communication Website of Aragón).

The prediction system arranges the categories, showing the ones which have more usage probability to the user in every moment. Inside every category, the pictograms can be chosen (these are ordered by usage probability too) adding a new word to the message in process. The prediction algorithm allows the user to pick his/her most common sentences to start the conversation, just like when he/she sends or receives a new message in order to do a more fluent talk. If the user decides to write a new sentence, the prediction system will help him/her to complete it, updating the suggestions every time he/she puts a new word or pictogram.

Related to the prediction system, it is based on N-grams language model. The N-grams are word sequences which length is N, and the prediction is done due to every N-gram probability at the language model. Also, the algorithm uses a global language model and a cache language model, which is based on the usage of every user. So, the cache language model personalize the system.

Finally, it is observed that the system performance is improved thanks to the cache according to the results obtained. When only the global language model is used by the algorithm, 9.15 % of the words are succesfully predicted and 1.10 words are introduced by keystroke (that is, every time a user of the dialogue system would employ the suggested prediction, he/she would add 1.10 words to the sentence which is being written). Also, it gets 8.37 % precision and 14.85 % recall. These results demonstrate the low quality of the global language model.

However, when only the cache language model is used, a better results are obtained. It reaches 84.42 % right words, 3.57 words per keystroke, 61.15 % precision and 67.91 % recall.

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Contexto.

El presente trabajo fin de máster se ha realizado dentro del grupo de investigación ViVoLab (del inglés *Voice Input Voice Output Laboratory*), perteneciente al departamento de Tecnologías de la Información y la Comunicación del Instituto de Investigación en Ingeniería de Aragón (I3A).

Además, este trabajo se enmarca dentro del proyecto IRIS [1], cuyo principal objetivo es proporcionar un herramienta de comunicación natural interactiva, accesible y adaptada a todo tipo de usuarios. Esta herramienta utiliza, para llevar a cabo la interacción hombre-máquina, los principios del diseño universal y de las interfaces de usuario naturales. Estos principios incluyen habla, gestos, dispositivos táctiles, pictogramas o voces sintéticas personalizadas.

1.2. Motivación y antecedentes.

La comunicación es una actividad básica del ser humano, que le permite relacionarse con su entorno. Para que esta comunicación sea efectiva debe realizarse bajo unas condiciones adecuadas. En concreto, para personas con problemas en la comunicación oral conseguir estas condiciones resulta más complicado.

En Aragón, el número de personas de 6 o más años de edad con algún tipo de discapacidad en el año 2008 era de 111,9 mil (9.19 % del total) [2]. A su vez, en España, la población afectada en ese mismo año era de 3.85 millones (8.55 % del censo) [3] ó 3.79 millones (8.97 % de los habitantes) si se habla únicamente de personas con 6 o más años de edad [2]. De esta forma, el número total de hogares con al menos una persona discapacitada asciende a 3.28 millones o 19.96 % de los mismos [3]. En concreto, y distinguiendo por el grupo de discapacidades que afectan a la relación del ser humano con su entorno, 734.2 mil personas (1.74 % de la población total o 19.39 % de la población que sufre algún tipo de discapacidad) tiene afectada la comunicación [2] y existen 621.2 mil personas (1.47 % ó 16.40 % respectivamente) cuya discapacidad está relacionada con las interacciones y relaciones personales [2].

Por otro lado, se estima que la tasa de palabras por minuto en la comunicación oral está entre 150 y 250, mientras que un experto mecanógrafo puede llegar a las 100 palabras/minuto y un usuario con discapacidad aproximadamente escribe 15 palabras/minuto [4] [5]. Por tanto, las

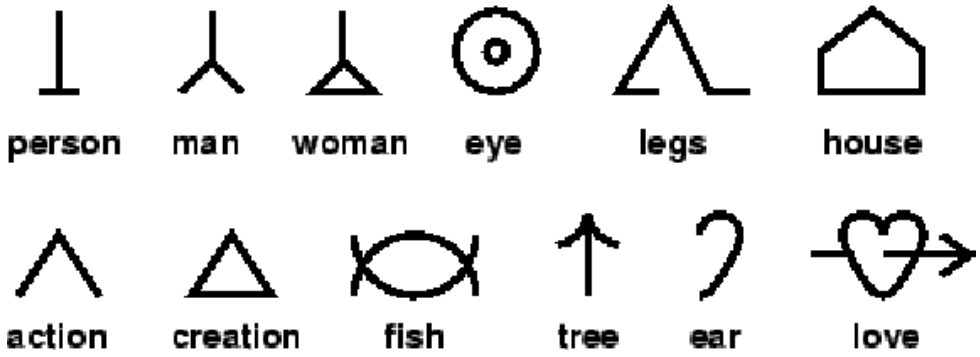


Figura 1.1: Ejemplos de símbolos Bliss.

personas que no son capaces de comunicarse completamente con el habla, ya sea por la falta de habilidad a la hora de hablar o porque no pueden expresar todas las funciones de la comunicación, necesitan de un sistema complementario o sustitutivo del habla, es decir, requieren de una comunicación aumentativa y alternativa (CAA o AAC -por sus siglas en inglés, de *Augmentative and Alternative Communication*-) para poder relacionarse con su entorno de una manera más efectiva y fluida. Las técnicas de CAA permiten el desarrollo del lenguaje en general, son una ayuda para la comunicación funcional y facilitan una mejora en el habla [6] [7].

De esta forma, para realizar una comunicación más interactiva se utilizan sistemas gráficos. Uno de los más comunes son los pictogramas. Los pictogramas son imágenes asociadas a palabras y permiten la comunicación de una manera más visual, constituyendo un tipo de lenguaje. La comunicación por pictogramas puede ser usada por personas con deficiencias en el habla [6] [8]. Otros sistemas de comunicación gráfica son PCS (del inglés, *Picture Communication Symbols*), el cual está formado por caracteres que incluyen dibujos simples y su palabra asociada (muy similar a los pictogramas); *Makaton Symbols*, que incluye tanto símbolos como signos manuales asociados a cada concepto [6] o el sistema de símbolos *Bliss*, el cual constituye un lenguaje gráfico basado en la semántica de las palabras [9]. Los símbolos *Bliss* (Figura 1.1) se crean a partir de la composición de iconos atómicos y están estructurados de manera que su representación otorgue conectividad a las palabras [10].

Además, entre las técnicas más usadas en CAA se encuentra la predicción de textos que ayuda a incrementar la ratio de palabras por minuto que un usuario puede introducir en el sistema [4]. Entre la predicción de textos, puede encontrarse la predicción de palabras a partir de los caracteres iniciales [4] o la predicción de frases al introducir las primeras palabras de las mismas [11]. Para mejorar esta predicción, se puede utilizar información de contexto, el cual puede venir dado por la propia conversación o a partir de información externa como por ejemplo día de la semana y hora o información de GPS [11] [12].

En este trabajo se desarrolla un sistema de comunicación basado en pictogramas, el cual incluye una predicción de frases incorporando información de contexto e información de caché relativa a cada usuario del sistema.

Así, la tarea de predicción de frases necesita de un modelo de lenguaje sobre el que asentarse. Con un buen modelo de lenguaje que utiliza procesado del lenguaje natural (NLP, del inglés, *Natural Language Processing*) es posible ahorrar pulsaciones de teclado debido a las sugerencias que el sistema realiza para seguir escribiendo [13]. A su vez, el objetivo principal del *NLP* es la generación automatizada de lenguaje natural (NLG, del inglés, *Natural Language Generation*)

del ser humano [14]. Entre los modelos de lenguaje más destacados se encuentran los modelos estocásticos basados en N-gramas y los basados en ejemplos. Los primeros requieren de algoritmos Viterbi para llevar a cabo la tarea de decodificación [13].

Los modelos de N-gramas están basado en la suposición de Markov, que indica que únicamente el contenido local anterior afecta a la siguiente palabra, es decir, la probabilidad de aparición de una palabra viene dada por las *(n-1)* palabras anteriores [14]. Estas probabilidades son calculadas a partir de la recopilación de textos de entrenamiento, los cuales constituirán un mejor modelo de lenguaje cuanto mejor se ajusten al contexto en el que se vaya a usar el sistema de predicción [14]. Asimismo, los modelos de N-gramas construidos a partir de la caché que el propio usuario va generando, componen un modelo de lenguaje que define la variación de la frecuencia de aparición de las palabras a corto plazo, y hace que la probabilidad de que una palabra vuelva a aparecer en un contexto en el que ya lo ha hecho recientemente sea mayor que en uno en el que no lo ha hecho [15].

Otro modelo de lenguaje es el de Ng-gramas, el cual es similar al de N-gramas pero añadiendo información gramatical basándose en la situación de la palabra en el texto (POS, del inglés, *Part Of Speech*), el cual asigna una probabilidad diferente dependiendo del tipo de palabra, como puede ser sustantivo, verbo o adjetivo, ya que según la situación en la frase será mas probable un tipo de palabra u otro [15] [16].

En cuanto a los modelos basados en ejemplos, el modelo es generado utilizando únicamente ejemplos específicos y no a partir de abstracciones derivadas de estos ejemplos, como puede ser el caso de los N-gramas. Todos estos ejemplos son descritos por el mismo conjunto de N atributos, lo cual permite categorizar aquellos ejemplos que compartan una serie de características. Utilizando, inicialmente, *funciones de similitud* entre cada una de las categorías existentes en el modelo de lenguaje y cada uno de los casos para los que queramos realizar una predicción, y posteriormente, *funciones de clasificación*, se elige la categoría que más se asemeja al caso bajo estudio. [17]

Otros modelos de lenguaje son los basados en redes neuronales. Hay principalmente dos tipos de redes neuronales utilizadas para construir modelos de lenguajes: las alimentadas hacia delante y las recurrentes. Las primeras utilizan la información de las *(n-1)* palabras anteriores para codificar un vector de dimensión $1xV$, siendo V la longitud del vocabulario (otorgará un *1* a las palabras que formen el (n-1)-grama y *0* en caso contrario). Este vector es proyectado linealmente a través de la matriz de proyección P del modelo de lenguaje. Por último, a partir de una capa oculta descrita por una función de activación (normalmente tangente hiperbólica o sigmoide logística) evalúa las probabilidades de todas las palabras del vocabulario de ser la palabra n de la secuencia dadas las *(n-1)* anteriores. Por otro lado, las redes neuronales recurrentes se diferencian en que la capa oculta toma información de toda la historia previa y no sólo de las *(n-1)* palabras anteriores, es decir, calcula la probabilidad de que cualquier palabra del vocabulario complete toda la historia de palabras anterior (para definir el vector $1xV$ se toma la información del (n-1)-grama anterior tal y como se hacía en las redes neuronales de alimentación hacia delante) . De esta forma, se pueden representar así modelos de contextos más amplios pero el coste computacional es más elevado. [18]

En cuanto a los algoritmos de predicción, algunos de los más usuales son los de tipo Viterbi, como son el *Suffix Tree* o el *Fussy Tree*. Estos algoritmos buscan completar frases evaluando palabra a palabra del vocabulario todos los caminos hasta que encuentran un fin de frase (como puede ser un punto o un signo de interrogación o exclamación), dando como resultado la frase o las frases más probables. La diferencia entre el *Suffix Tree* y el *Fussy Tree* es que éste último

evalúa todos los caminos posibles mientras que el primero poda en cada paso aquellos cuya probabilidad es muy reducida [19]. Por tanto, el *Fussy Tree* da unas mejores predicciones a costa de un coste computacional mayor.

Otro algoritmo utilizado para la búsqueda de frases es el TF-IDF (del inglés, *Term-Frequency - Inverse Document Frequency*). Este algoritmo analiza la similitud entre una frase completa y una secuencia de palabras de la misma longitud que contenga la palabra a analizar, calculando así la probabilidad de que cada una de las palabras del vocabulario complete una secuencia [20].

Los algoritmos analizados anteriormente pueden dar lugar a una o varias predicciones, creando un *top-k* o *ranked-suggestions*, es decir, las propuestas se ordenan de mayor a menor probabilidad de completar la frase en cuestión [19].

Finalmente, para valorar las prestaciones de un sistema de predicción en un sistema de comunicaciones hay dos conceptos que son de principal importancia: elocuencia y fluidez. La elocuencia indica el grado en que la mejor de las opciones propuestas encaja con lo que el usuario de verdad quiere decir, mientras que la fluidez está relacionada con la tasa de comunicación, es decir, el número de caracteres producidos por unidad de tiempo. Estos dos conceptos son opuestos, así que hay que alcanzar un compromiso entre ambos para contar con un sistema con buenas prestaciones [16].

Así, la elocuencia es evaluada a partir de los conceptos de *precisión* y *recall*. *Grabski y Scheffer* [20] evalúan la *precisión* como el cociente entre las sugerencias aceptadas y el número de peticiones hechas; el *recall* lo calcula como la relación entre las sugerencias aceptadas y las realizadas.

1.3. Objetivos del trabajo fin de máster.

El objetivo principal de este trabajo fin de máster es el desarrollo de un módulo de diálogo que permita la comunicación funcional mediante la utilización de texto y pictogramas entre personas con problemas en la comunicación oral. Este módulo constará de los siguientes elementos:

1. Una aplicación tipo chat, llamada TolkiChat, que incluye una interfaz gráfica de usuario (GUI, del inglés *Graphical User Interface*).

2. Predicción de frases utilizando modelos de lenguaje basados en N-gramas, donde se cuantifica la probabilidad de que aparezca una palabra basada en la secuencia de *(n-1)* palabras anteriores, y modelos de lenguaje tipo caché para personalizar el sistema. La predicción de frases se realiza a partir de un algoritmo tipo Viterbi-*Suffix Tree*.

1.4. Organización.

A continuación, se indica la distribución que va a seguir el trabajo a partir de ahora:

- **Capítulo 2: *Comunicación Aumentativa y Alternativa (CAA) y sistemas de diálogo.*** Revisión bibliográfica de los sistemas de diálogo que utilizan CAA existentes.

- **Capítulo 3:** ***Modelos de lenguaje y aprendizaje predictivo.*** En esta parte se explica la base teórica sobre la que se asienta el trabajo. En primer lugar, se describen los modelos de lenguaje utilizados, tanto el global como los relacionados con la personalización del sistema, es decir, la caché, para, posteriormente, definir los algoritmos creados para realizar la predicción de frases.
- **Capítulo 4:** ***Bases de datos.*** Se detallan las bases de datos que se utilizan a lo largo de este trabajo, tanto la relacionada a la obtención de pictogramas como las utilizadas para construir los modelos de lenguaje.
- **Capítulo 5:** ***TolkiChat.*** Desarrollo del proceso seguido para la creación de la aplicación, llamada TolkiChat y que supone el producto final del presente trabajo. También se razona la ordenación de los datos en la misma.
- **Capítulo 6:** ***Resultados y Discusión.*** Aquí se muestran los resultados del sistema creado y su capacidad para predecir frases y aumentar la tasa de palabras por pulsación que el usuario puede introducir, interpretándolos y comparándolos con la base teórica. Además, se señalan las fortalezas y debilidades de la aplicación construida.
- **Capítulo 7:** ***Conclusiones.*** En esta sección se recopilan las conclusiones y las aportaciones realizadas.
- **Capítulo 8:** ***Líneas futuras.*** Se enumeran las posibles extensiones del trabajo.

Capítulo 2

Comunicación Aumentativa y Alternativa (CAA) y sistemas de diálogo

La comunicación es una actividad dinámica e interpersonal que necesita principalmente de habilidades en cuatro áreas interrelacionadas. Estas áreas son competencia lingüística, competencia operacional, competencia social y competencia estratégica [21]. Sin embargo, esta definición puede diferir para aquellas personas con dificultades en el desarrollo del lenguaje verbal y la comunicación, las cuales necesitan trabajar de manera simultánea la integración sensorial, la comunicación y la salud emocional para desarrollar estas habilidades de manera rápida y efectiva [22]. Para ello, se integra la tecnología en los procesos de comunicación, es decir, se realiza una comunicación aumentativa y alternativa (CAA).

La comunicación aumentativa y alternativa incluye todas las formas de comunicación que son usadas para expresar pensamientos, necesidades, deseos e ideas [23]. Para que el uso de los sistemas de CAA alcancen la competencia comunicativa deben ser útiles respecto a las demandas de la rutina diaria, deben ajustarse al contexto de la comunicación y deben conseguir que el individuo pueda adecuarse, a partir del uso del lenguaje del sistema, a las reglas sociales comúnmente usadas al comunicarse en cualquier idioma [21].

Así, para alcanzar las competencias mencionadas en el párrafo anterior, las técnicas más utilizadas por los sistemas CAA son síntesis del habla y construcción de frases siguiendo la conversación en curso [24]. Las limitaciones de estos procesos son la velocidad para adecuarse al ritmo normal de una conversación, lo cual puede llevar a una interacción forzada y poco natural [24] [25]. La primera de las limitaciones puede resolverse mediante una mejora de la interfaz de usuario o mediante la reducción de los datos de entrada que necesita el sistema [25]; para la segunda se hace uso del procesado del lenguaje natural (NLP), el cual permite mejorar la usabilidad y productividad del sistema en el que es implementado [24].

Uno de los sistemas de diálogo que intenta incrementar la velocidad de la conversación es expuesto por *Copestake* [25]. Se trata de un sistema de comunicación por texto, y basa su rapidez en reducir el número de pulsaciones de teclado que el usuario introduce en el sistema. Para ello, utiliza predicciones de caracteres para completar palabras y de palabras para completar oraciones. Esta predicción se realiza a partir de la probabilidad de ocurrencia de cada palabra a partir del

uso previo del propio usuario. Además, se realiza una mejora de la predicción a partir de la información POS, de gramática y de contexto.

Por otro lado, el sistema descrito por *Dempster et al.* [24] hace uso de NLP y NLG para dotarlo de un lenguaje más natural. Este sistema genera el lenguaje a partir de ontologías, es decir, construye modelos lógicos y jerárquicos dadas las relaciones que tienen diferentes conceptos en un dominio determinado. Asimismo, utiliza árboles sintácticos para ordenar las palabras construyendo frases, como se puede ver en la Figura 2.1.

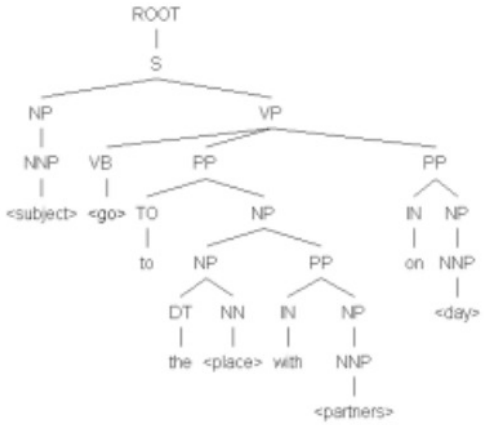


Figura 2.1: Árbol sintáctico utilizado por Dempster et al. [24] en su sistema de NLG.

Además, el desarrollo de la comunicación y el lenguaje es más efectivo si se apoya con estímulos visuales, como pueden ser pictogramas, PCS o *Makaton Symbols*, los cuales se han descrito en el Capítulo 1. Nos centraremos a partir de ahora, en los pictogramas ya que son los utilizados en este trabajo, y, en general, los más comúnmente usados.

Se ha demostrado que los pictogramas ayudan a la comprensión del mensaje y a la adquisición del conocimiento transmitido en la comunicación, siendo comprendidos, además, independientemente del idioma que cada persona utilice para comunicarse [26] [27]. En concreto, son muy efectivos para acompañar mensajes de advertencia y peligro, ya que el mensaje es captado de una forma más evidente y significativa [27].

Así pues, un sistema CAA basado en imágenes es el descrito por *Baker* [28], llamado Minspeak™, el cual a partir de un conjunto reducido de imágenes construye frases (Figura 2.2). Se basa en que cada imagen puede aportar distintos significados. Por ejemplo, una imagen de una cama puede referirse propiamente a dicho mueble o puede describir la acción de dormir, si esa imagen va acompañada de otra en el que se muestra una persona moviéndose [28] [29].

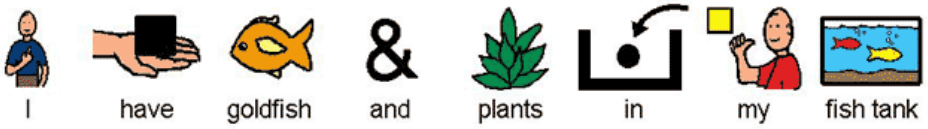


Figura 2.2: Traducción de la frase "Tengo un pez dorado y plantas en mi pecera" al sistema Minspeak™.

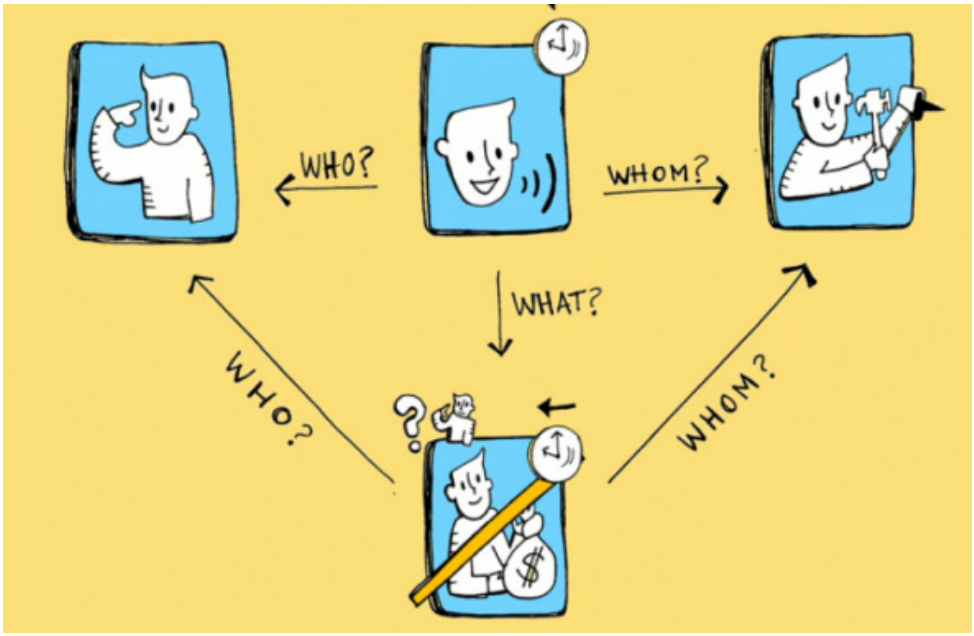


Figura 2.3: Ejemplo de construcción de la frase "Le dije al carpintero que no podía pagarle" en el sistema FreeSpeech^{TM}.

Otro sistema CAA basado en pictogramas es *FreeSpeech*^{TM}, explicado por *VocalID*^{TM}*et al.* [29]. Esta aplicación desarrolla mapas de imágenes, basándose en las respuestas a preguntas clave como pueden ser *¿Qué?*, *¿Cuándo?*, *¿Dónde?*, *¿Quién?*, etc. A partir de estas preguntas, se relacionan diferentes palabras, para, finalmente, construir oraciones. Asimismo, se usan filtros para añadir otras connotaciones extra a la frase como negación o tiempo en el que se desarrolla la acción (relacionado con la conjugación de los verbos). En la Figura 2.3 se puede observar cómo se construye una frase en la aplicación *FreeSpeech*^{TM}. En concreto: *"Le dije al carpintero que no podía pagarle".*

Finalmente, otra aplicación muy sencilla que construye oraciones con pictogramas es *Pikto-Plus*, la cual, simplemente, asocia un pictograma a cada palabra para ayudar al desarrollo del lenguaje de niños y personas con discapacidad [30].

Capítulo 3

Modelos de lenguaje y aprendizaje predictivo

Tras el repaso inicial a los modelos de lenguaje utilizados para realizar predicción hecho en el Capítulo 1, a continuación se va a explicar más en detalle el utilizado en este trabajo.

Así, el modelo de lenguaje que se ha utilizado es el basado en N-gramas. Esto se debe a la simplicidad y flexibilidad del modelo y a la posibilidad de introducir nuevos elementos al vocabulario, algo importante cuando uno de los objetivos del trabajo es la implementación de una aplicación tipo chat. Esto implica que van a surgir durante la comunicación palabras no contempladas en el vocabulario, como pueden ser nombres propios. Con otros modelos, como el realizado con redes neuronales no es evidente incorporar nuevas palabras, ya que se necesitaría entrenar la red de nuevo cada vez que se añade un nuevo componente.

En cuanto al modelo basado en ejemplos, otro de los mencionados en el Capítulo 1, su alcance es más reducido al basarse en frases concretas y no en abstracciones de las mismas como sí realiza el modelo de N-gramas, lo cual lo dota de una mayor flexibilidad. Por tanto, el modelo basado en ejemplos puede ser útil para complementar otro modelo pero no para ser el modelo de lenguaje principal.

3.1. N-gramas.

Un N-grama se trata de una secuencia de palabras de longitud N (Ecuación 3.1). Su probabilidad se calcula de acuerdo a la Ecuación 3.2. La aproximación de la Ecuación 3.3 define el modelo de N-gramas, en el que la palabra k depende de las N-1 anteriores [31].

$$\omega_1^n = \{\omega_1, \omega_2, ..., \omega_n\} \tag{3.1}$$

$$P(\omega_1^n) = P(\omega_1)P(\omega_2|\omega_1)P(\omega_3|\omega_1^2)\cdots P(\omega_n|\omega_1^{n-1}) = \prod_{k=1}^{n} P(\omega_k|\omega_1^{k-1}) \tag{3.2}$$

$$P(\omega_1^n) \cong \prod_{k=1}^{n} P(\omega_k|\omega_{k-N+1}^{k-1}) \tag{3.3}$$

Así, el modelo de lenguaje de N-gramas se basa en calcular la probabilidad de la palabra n dadas las *(n-1)* anteriores. Esta probabilidad se estima con un estimador MLE de máxima verosimilitud (del inglés, *Maximum Likelihood Estimation*). Esto es, se calcula el cociente entre el número de veces que ha tenido lugar el N-grama $\omega_1, \omega_2, ..., \omega_{n-1}, \omega_n$ y el sumatorio del número total de veces de ese mismo (N-1)-grama tal y como se muestra en la Ecuación 3.4.

$$p = P(\omega_n|\omega_1^{n-1}) = \frac{conteo(\omega_1^n)}{conteo(\omega_1^{n-1})} \tag{3.4}$$

3.2. Skip-gramas.

Además de los N-gramas, para obtener secuencias de palabras que no se han dado pero que pueden ser altamente probables se usan los Skip-gramas. Los Skip-gramas calculan la probabilidad de una palabra a partir no de la inmediatamente anterior, si no descartando K palabras entre una palabra y otra. Cuanto mayor sea K y mayor sea el tamaño del Skip-grama menos relación guardarán las palabras de la secuencia entre sí, siendo menos relevante su probabilidad.

La probabilidad de un Skip-grama se calcula con la expresión 3.5.

$$P_{skip-gram} = P(\omega_n|\omega_{n-(K+1)}, \omega_{n-(2*K+1)}, ..., \omega_{n-(N*K+1)}) \tag{3.5}$$

Concretamente, en este trabajo se ha utilizado K=1 y sólo bigramas como se muestra en la ecuación 3.6. Además, solo se tienen Skip-gramas provenientes de la caché y no del modelo de lenguaje global (ver secciones 3.4 y 3.5).

$$P_{skip-gram} = P(\omega_n|\omega_{n-2}) \tag{3.6}$$

3.3. Perplejidad.

Una buena medida de calidad del modelo de lenguaje es la perplejidad. La perplejidad mide la incertidumbre del modelo, siendo menor cuanto mejor sea el modelo de forma que se pueden realizar predicciones con alta probabilidad de acierto. La perplejidad se calcula como la probabilidad inversa de un conjunto de elementos de test, normalizada por el número de palabras de este conjunto (Ecuación 3.7).

$$ppl = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(\omega_i|\omega_1^{i-1})}} = \left(\prod_{i=1}^{N} P(\omega_i|\omega_1^{i-1})\right)^{-\frac{1}{N}} \tag{3.7}$$

La perplejidad está relacionada con el número de alternativas que existan para seguir una secuencia [32], siendo mayor cuanto más opciones existan, y, por tanto, más difícil de realizar una predicción acertada. Por ejemplo, para la secuencia *'Ocurrió en el mes de'* existen 12 posibilidades. Si todas son equiprobables tenemos:

$$P(\omega_i|\text{Ocurrió en el mes de}) = P(\text{enero}|\text{Ocurrió en el mes de}) = \\ = P(\text{febrero}|\text{Ocurrió en el mes de}) = \cdots = P(\text{diciembre}|\text{Ocurrió en el mes de}) = \frac{1}{12} \tag{3.8}$$

Y su perplejidad se calcula según la Ecuación 3.9.

$$ppl = \left(\prod_{i=1}^{N} P(\omega_i|\text{Ocurrió en el mes de})\right)^{-\frac{1}{N}} = \left(\frac{1}{12}^{N}\right)^{-\frac{1}{N}} = 12 \tag{3.9}$$

Siguiendo el mismo procedimiento para el ejemplo *'Su nombre empieza por la letra'* en las ecuaciones 3.10 y 3.11, se observa como la perplejidad es mayor en este caso y por tanto más difícil de predecir.

$$P(\omega_i|\text{Su nombre empieza por la letra}) = P(\text{A}|\text{Su nombre empieza por la letra}) = \\ = P(\text{B}|\text{Su nombre empieza por la letra}) = \cdots = P(\text{Z}|\text{Su nombre empieza por la letra}) = \frac{1}{27} \tag{3.10}$$

$$ppl = \left(\prod_{i=1}^{N} P(\omega_i|\text{Ocurrió en el mes de})\right)^{-\frac{1}{N}} = \left(\frac{1}{12}^{N}\right)^{-\frac{1}{N}} = 27 \tag{3.11}$$

3.4. Caché.

Las frases utilizadas por el usuario se almacenan en la caché, siendo ésta personalizada y adaptada a cada usuario. Cada vez que una frase es introducida, se divide en unigramas, bigramas... llegando hasta N-gramas siendo N el tamaño máximo de la secuencia de palabras a almacenar. En este trabajo se ha utilizado N = 5.

Para cada N-grama se calcula su probabilidad y se actualiza la del resto de N-gramas almacenados en caché de la siguiente forma:

1. Se suma 1 al denominador de todos los N-gramas que compartan el (N-1)-grama ω_{n-N+1}^{n-1}. Si el N-grama es nuevo, su denominador será el mismo que el de cualquier otro N-grama que comparta el mismo (N-1)-grama. Si el (N-1)-grama también es nuevo, se inicializa el denominador a 1.

2. Se suma 1 al numerador del N-grama ω_{n-N+1}^{n}. Si es la primera vez que este N-grama ocurre se inicializa a 1.

   - Si ω_n no está en el vocabulario, se añade.

3. Se recalculan todas las probabilidades de los N-gramas afectados en los pasos anteriores dividiendo el numerador entre el denominador actualizados.

3.5. Probabilidad total.

Tras lo explicado en las secciones anteriores, la probabilidad total de un N-grama se calcula teniendo en cuenta su probabilidad en el modelo global, en la caché y, también, los skip-gramas. En la ecuación 3.12 se define esta probabilidad total.

$$p_{total} = (1 - \alpha_{skip}) * (\alpha_{cache} * p_{cache} + (1 - \alpha_{cache}) * p) + \alpha_{skip} * p_{skip-gram} \tag{3.12}$$

La caché se tiene en cuenta cuando empieza a ser relevante, esto es cuando consta de más de M palabras. Si la caché no es relevante o no existe el N-grama bajo evaluación en la caché, la probabilidad total es la mostrada en la ecuación 3.13. Tampoco se tendrá en cuenta en este caso la probabilidad del skip-grama, ya que en este trabajo solo se almacenan skip-gramas en la caché y no entre los N-gramas globales. Por tanto, si la caché no es relevante tampoco lo será la del skip-grama.

$$p_{total} = p \tag{3.13}$$

En cambio, si una palabra del N-grama no existe en el vocabulario inicial, su probabilidad total se tiene en cuenta sin tener en cuenta la probabilidad global tal y como muestra la ecuación 3.14.

$$p_{total} = (1 - \alpha_{skip}) * p_{cache} + \alpha_{skip} * p_{skip-gram} \tag{3.14}$$

Lo mismo ocurre con la probabilidad del skip-grama. Si el N-grama evaluado no está entre los skip-gramas, la probabilidad total es la determinada por la ecuación 3.15.

$$p_{total} = \alpha_{cache} * p_{cache} + (1 - \alpha_{cache}) * p \tag{3.15}$$

Además, si alguna de las palabras del N-grama no está en el vocabulario inicial, la probabilidad total será la misma que la probabilidad del N-grama en la caché directamente. Esto se muestra en la expresión 3.16.

$$p_{total} = p_{cache} \tag{3.16}$$

3.6. Predicción

La predicción se logra a partir del conjunto de palabras que obtiene máxima probabilidad, es decir:

$$\arg\max_{\omega_n} p = \arg\max_{\omega_n} P(\omega_n|\omega_1^{n-1}) \tag{3.17}$$

Generalizando el modelo, la predicción de K palabras a partir de las (n-1) anteriores sería la mostrada en la ecuación 3.18. En la expresión 3.19 se factoriza la probabilidad conjunta de

todas las palabras desconocidas; mostrando la suposición de Markov de orden N que sustenta la simplificación del modelo de N-gramas en la ecuación 3.20 [13].

$$\arg\max_{\omega_n^{n+K-1}} P(\omega_n^{n+K-1}|\omega_1^{n-1}) \tag{3.18}$$

$$= \arg\max_{\omega_n^{n+K-1}} \prod_{i=0}^{K-1} P(\omega_{n+i}|\omega_1^{n+i-1}) \tag{3.19}$$

$$= \arg\max \prod_{i=0}^{K-1} P(\omega_{n+i}|\omega_{n+i-N+1}^{n+i-1}) \tag{3.20}$$

Además de esta probabilidad, en los N-gramas se ha aplicado un índice corrector llamado *backoff*, el cual se aplica cuando no existe el N-grama seleccionado y se tiene que ir evaluando el *(N-1)-grama*, *(N-2)-grama*... sucesivamente hasta que se encuentra uno existente, pudiendo llegar hasta el unigrama. Si finalmente es evaluado el *(N-K)-grama*, el *backoff* se evalúa en el *(K+1)-grama* siendo K el número de palabras descartadas de la evaluadas del *(N-K)-grama*. El *backoff* es mejor para aquellos *(N-K)-gramas* que históricamente sean más probables.

Por ejemplo, si queremos evaluar el 5-grama *'Quiero comer en media hora'* y no existe dicho 5-grama ni el 4-grama *'comer en media hora'* pero sí el trigrama *'en media hora'*, la probabilidad final sería la mostrada en la ecuación 3.21.

$$P(\text{Quiero comer en media hora}) = P(\text{hora} \mid \text{en media}) + P_{backoff}(\text{quiero comer en}) \tag{3.21}$$

El parámetro de *backoff* es calculado mediante el método de Kneser-Ney [33] en el modelo de lenguaje global. Este modelo se basa en la idea que el parámetro de *backoff* es relevante en un N-grama únicamente cuando su probabilidad es baja o nula. Por ejemplo, el trigrama 'Santiago de Compostela' puede ser común pero no tanto el bigrama *'de Compostela'*. Por tanto, es adecuado dar una baja probabilidad a dicho bigrama pero potenciar, a partir de su parámetro de *backoff*, su aparición tras *'Santiago'*.

Por otro lado, para el modelo de lenguaje caché, el backoff se calcula mediante el método de Katz (ver Anexo A).

3.7. Algoritmo Viterbi.

El algoritmo Viterbi es un algoritmo de programación dinámica que permite hallar los caminos más probables (en este caso, haciendo uso de los N-gramas explicados en el apartado 3.1) dada una secuencia inicial. Aquí, la secuencia inicial será una palabra o conjunto de palabras con las que se predicen las restantes para completar la frase.

De esta forma, en la primera recursión ($s = 0$) del algoritmo Viterbi se recorren todas las palabras ω_k pertenecientes al vocabulario para formar N-gramas junto con las *N-1* últimas palabras que componen la secuencia inicial . Todos estos N-gramas son evaluados, de manera que si la probabilidad del N-grama más probable es menor al umbral θ_1, el algoritmo finaliza.

En caso contrario, todos los N-gramas con probabilidad mayor a otro umbral θ_2 se almacenan (podando el resto) y forman los caminos sobre los que ejecutar de nuevo el algoritmo Viterbi, excepto si el N-grama de mayor probabilidad acaba en un signo de puntuación que indique un fin de frase (es decir, punto o signos de interrogación o exclamación). En ese caso, también finalizará el algoritmo. Además, si algún camino activo aún no siendo el mejor acaba en un símbolo de fin de frase, se guarda como posible resultado una vez que el algoritmo termine, no siguiendo evaluando el algoritmo sobre él.

Si el algoritmo aún no ha finalizado, se vuelven a evaluar nuevos N-gramas tras incrementar el paso s, recorriendo todo el vocabulario de nuevo sobre todos los caminos que permanecen activos. Así, se vuelven a realizar todos los pasos anteriores hasta que se dé una condición de finalización. Para terminar, el resultado de la ejecución de este algoritmo serán las secuencias de longitud s compuestas por las palabras que constituyen los caminos que estaban activos o guardados, es decir, ω_n^{n+i}, siendo i la longitud de cada frase guardada o que pertenecía a un camino activo.

Por otro lado, los umbrales θ_1 y θ_2 son adaptativos. Así, si la probabilidad del mejor N-grama aumenta respecto a los umbrales, éstos también aumentarán; en caso contrario, los umbrales disminuirán.

A continuación, en la tabla 3.1 se resume el algoritmo Viterbi diseñado en este trabajo y que se acaba de explicar.

Antes de eso, remarcar algunos de los términos que se van a usar en el algoritmo:

- $\underline{\boldsymbol{\omega_{n-N+1}^{n}}}$**:** Últimas N palabras consecutivas de una secuencia. Por ejemplo, si la secuencia de palabras es *'Yo voy a comprar el pan todos'* y N = 5:

$$\omega_{n-N+1}^{n} = \text{'a comprar el pan todos'}$$

- **s:** Número de veces que se ha recorrido el algoritmo Viterbi. Indica, además, el número de palabras de las que constan las predicciones activas.

- $\underline{\boldsymbol{\omega_{n+s-N+1}^{n+s}}}$**:** Últimas N palabras consecutivas de la unión de la secuencia inicial con una predicción activa. Por ejemplo, si la serie inicial es *'Yo voy a comprar el pan todos '* con N = 5, s = 2 y predicción *'los domingos'*:

$$\omega_{n+s-N+1}^{n+s} = \text{'el pan todos los domingos'}$$

- $\underline{\boldsymbol{\theta_1}}$**:** Umbral de parada del algoritmo Viterbi. Se actualiza, a posteriori, con la probabilidad $\max P(\omega_k|\omega_{n+s-N+1}^{n+s-1})$, recorriendo ω_k sobre todo el vocabulario.

- $\underline{\boldsymbol{\theta_2}}$**:** Umbral de poda de caminos. Al igual que θ_1, se actualiza de acuerdo a la probabilidad $\max P(\omega_k|\omega_{n+s-N+1}^{n+s-1})$. Aunque éste lo hace antes de ser usado en el paso actual.

1. Partiendo de la secuencia ω_1^{n-1}, la secuencia inicial de entrada al algoritmo Viterbi será el (N-1)-grama ω_{n-N+1}^{n-1}. Además, se inicializa el paso $s = 0$.

2. Se ejecuta el algoritmo Viterbi recursivamente hasta que haya un **break**:

   a) **For**. Para todos los caminos activos $\omega_{n+s-N+1}^{n+s-1}$ en el paso s, se recorre todo ω_k perteneciente al vocabulario calculando la probabilidad condicionada de ω_k dado el N-grama $\omega_{n+s-N+1}^{n+s-1}$. Esto es: $P(\omega_k|\omega_{n+s-N+1}^{n+s-1})$.

   b) **If** $\max P(\omega_k|\omega_{n+s-N+1}^{n+s-1}) < \theta_1$, entonces **break**.

   c) Actualizar los umbrales θ_1 y θ_2 según $\max P(\omega_k|\omega_{n+s-N+1}^{n+s-1})$.

   d) **If** $\max P(\omega_k|\omega_{n+s-N+1}^{n+s-1}) < \theta_2$ se poda el camino $\omega_{n+s-N+1}, ..., \omega_{n+s-1}, \omega_k$. En caso contrario, sigue activo pasando ω_k a ser ω_{n+s} si ω_k ! = (‘.’ or ‘?’ or ‘!’). Si ω_k == (‘.’ or ‘?’ or ‘!’), entonces se guarda $\omega_n, \omega_{n+1}, ..., \omega_{n+s} = \omega_k$ como posible resultado una vez que el algoritmo haya finalizado.

   e) **If** $\underset{\omega_k=\omega_{n+s}}{\arg\max} P(\omega_k = \omega_{n+s}|\omega_{n+s-N+1}^{n+s-1})$ == (‘.’ or ‘?’ or ‘!’), entonces **break**.

   f) Se incrementa s.

3. **Return**. Se devuelven como salida de la función los ω_n^{n+i} pertenecientes a los K mejores caminos que quedaban activos o guardados antes de realizar el **break**.

Tabla 3.1: Resumen del algoritmo Viterbi diseñado en este trabajo.

Para finalizar, en la figura 3.1 se muestra un ejemplo de aplicación del algoritmo Viterbi explicado anteriormente. A partir de la palabra inicial *‘IR’* se obtiene como frase más probable, y, por tanto, resultado de la predicción, *‘IR A CASA .’*. Los rombos negros contiene las probabilidades logarítmicas acumuladas en cada paso, podándose (señalado con un aspa roja) los caminos con menor probabilidad (la poda se basa en un umbral adaptativo que va variando en función de la probabilidad del mejor camino en cada paso) y guardándose aquellos (marcado con un círculo verde) cuya última palabra es un símbolo de fin de frase (en este ejemplo, un punto). El criterio de parada en este ejemplo es el definido por el punto 2e) de la tabla 3.1, es decir, que el mejor camino finaliza en un signo de puntuación de fin de frase (un punto en este caso). Una vez que el algoritmo ha finalizado, se devuelve como resultado el mejor camino activo y/o guardado: *‘IR A CASA.’*.

3.8. Conjugación de verbos.

Un caso especial son los verbos, ya que cada uno está compuesto por más de 100 formas verbales. Por tanto, se restaría eficacia en la predicción si se tuviera en cuenta cada una por separado en el modelo de lenguaje. Igualmente, como se verá en el Capítulo 4 (ver sección 4.1.1), sólo existen pictogramas asociados a los infinitivos de los verbos. Por ello, al resto de formas verbales se les asigna el pictograma de su infinitivo.

Para obtener una predicción eficaz pero respetando las conjugaciones a su vez, se realiza un post-procesado del la frase dada como resultado por parte del algoritmo Viterbi. Para ello, se

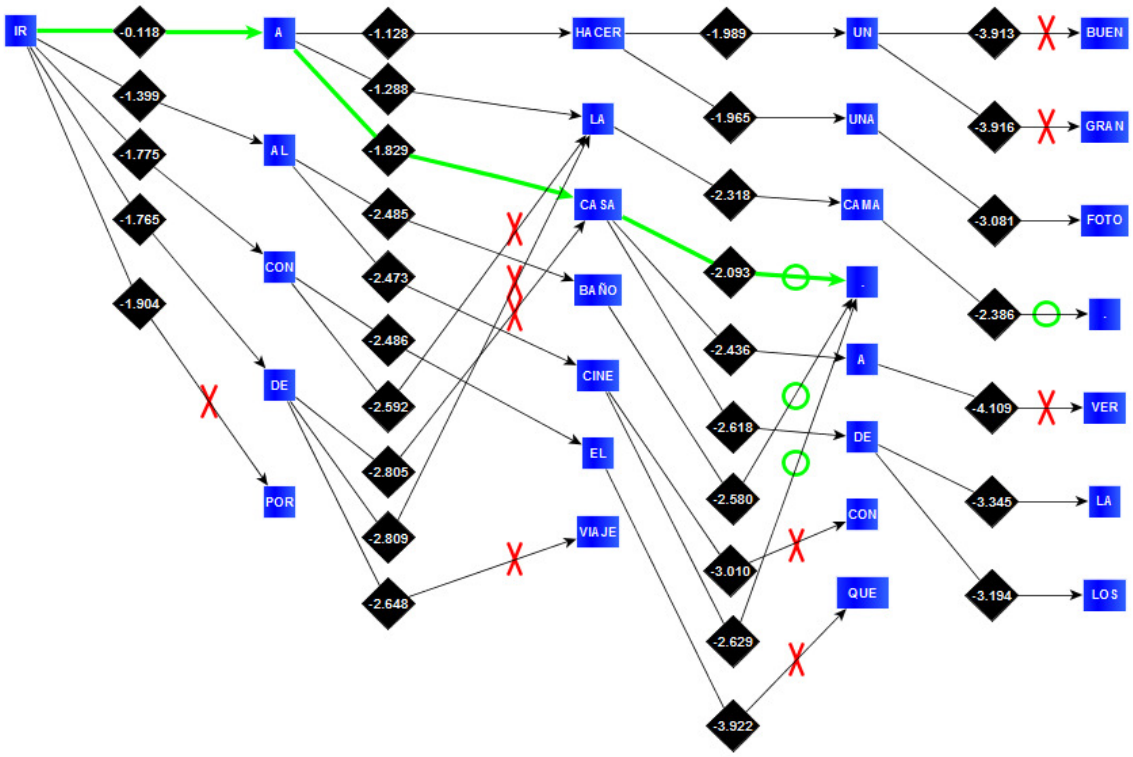


Figura 3.1: Ejemplo de funcionamiento del algoritmo Viterbi explicado. Predicción a partir de la palabra 'IR', obteniéndose como frase más probable 'IR A CASA .'.

disponen de dos modelos de lenguaje, uno que sólo consta de infinitivos y otro que contiene todas las formas verbales que aparecen en los textos que sirven como base de los modelos de lenguaje. Por tanto, los pasos que se efectúan en una frase que incluye verbos son los siguientes:

1. Se realiza la ejecución del algoritmo Viterbi sobre el modelo de lenguaje que únicamente contiene infinitivos. Por ejemplo, dando como resultado: *'Nosotros comprar leche .'*.

2. Se identifican todos los verbos que aparecen en la frase dada como resultado del paso anterior. Para cada verbo de la frase, se toman como referencia todas las palabras anteriores al mismo y se evalúan todas las formas verbales del verbo bajo análisis cambiando el infinitivo por aquella forma más probable. De esta manera, obtendríamos como resultado final: *'Nosotros compramos leche .'*.

   Además, una mejora del sistema sería utilizar no sólo las palabras anteriores, si no también las M palabras posteriores para realizar la conjugación de los verbos.

De la misma forma que ocurre con los modelos de lenguaje global, la caché también tendrá dos modelos de lenguaje equivalentes. Uno formado únicamente por infinitivos y otro que contiene todas las formas verbales.

Capítulo 4

Bases de datos

En el presente trabajo se utiliza una base de datos para adquirir los pictogramas asociados a cada palabra y otra base de datos para construir los modelos de lenguaje globales. A continuación, se explica cada una de ellas por separado.

4.1. Pictogramas: Base de datos Arasaac.

Esta base de datos es provista por Arasaac (Portal Aragonés de Comunicación Aumentativa y Alternativa). Arasaac es un Sistema Aumentativo y Alternativo de Comunicación (SAAC) basado en pictogramas que facilitan la comunicación de personas con dificultades. Este catálogo de pictogramas es propiedad del Gobierno de Aragón [34].

Arasaac nació en el año 2007 por la colaboración entre el CATEDU (Centro Aragonés de Tecnologías para la Educación) y el colegio público de educación especial Alborada, con la colaboración del Centro Politécnico Superior y finaciado por el Departamento de Industria e Innovación del Gobierno de Aragón [34].

Esta base de datos consta de 14013 palabras asociadas a 30028 pictogramas. Además, un total de 2995 pictogramas se han agrupado en 49 categorías. A partir de estos pictogramas categorizados, se ha definido un vocabulario básico, utilizado en este trabajo, de 4079 palabras, las cuales están asociadas a 3514 pictogramas. En la Tabla 4.1 se detallan las categorías de pictogramas incluidas la base de datos de Arasaac.

NOTA: La categoría 38 (*Pictogramas para categorías*) se encontraba no operativa durante la realización del presente proyecto y por tanto no ha sido utilizada.

Así pues, dicha base de datos es de tipo relacional y se utiliza el sistema de gestión mySQL para acceder a la información almacenada en la misma. En este caso, los pictogramas asociados a una palabra determinada o viceversa.

De esta forma, para acceder a la base de datos utilizaremos el siguiente comando:

*mysql -h dihana.cps.unizar.es -u *** - p ****

Indicando en *-h* el dominio donde se encuentra la base de datos, en *-u* el usuario y en *-p* la contraseña.

ID	Categoría	Número de pictogramas
1	**Alfabeto**	79
2	**Alimentos**	178
3	**Alternativo**	4
4	**Animales**	114
5	**Aparatos**	28
6	**Aseo**	58
7	**Ayudas Técnicas Discapacidad**	65
8	**Bebidas**	33
9	**Casa**	67
10	**Clima**	31
11	**Cocina**	50
12	**Colegio**	132
13	**Colores Formas Medidas**	78
14	**Conceptos Espaciales Básicos**	22
15	**Cualidades**	76
16	**Cuerpo**	123
17	**Deportes**	129
18	**Dinero**	25
19	**Enfermedad**	86
20	**Expresiones**	32
21	**Fiestas**	38
22	**Herramientas**	86
23	**Informática**	44
24	**Juegos**	45
25	**Juguetes**	42
26	**Limpieza**	31
27	**Lugares**	107
28	**Medicinas Pruebas Médicas**	69
29	**Medios de Comunicación**	38
30	**Muebles**	24
31	**Música**	49
32	**Natación**	28
33	**Navidad**	57
34	**Números**	11
35	**Partículas**	14
36	**Personajes**	59
37	**Personas**	52
38	**Pictogramas para categorías**	0
39	**Plantas**	82
40	**Posesivos**	12
41	**Preguntas**	75
42	**Preposiciones**	19
43	**Profesiones**	137
44	**Sentimientos Estados**	20
45	**Tiempo**	54
46	**Tiendas**	49
47	**Transportes**	65
48	**Verbos**	168
49	**Vestido**	110
Total		2995

Tabla 4.1: División de la base de datos de pictogramas de Arasaac por categorías.

Para elegir la base de datos, en nuestro caso Arasaac, realizamos la siguiente llamada:

use arasaac

De esta forma, si queremos obtener el pictograma de una palabra (en este caso, utilizaremos como ejemplo *'comer'*), realizamos la siguiente serie de peticiones a la base de datos:

1. Se halla el ID de la palabra correspondiente, en este caso *'comer'*:

   select id_palabra from palabras where palabra = 'comer';

   Lo cual arroja los siguientes resultados:

palabra	**id_palabra**
comer	1
comer	2836

Tabla 4.2: Asociación de la palabra 'comer' con sus IDs.

   En principio, se escoge el primer ID asociado a la palabra. Si al finalizar todo el proceso de obtención del pictograma no obtuviéramos uno, se realizaría de nuevo el proceso a partir de aquí eligiendo el siguiente ID, y así sucesivamente hasta la obtención de un pictograma o hasta que se agotaran los IDs.

2. Se adquiere el ID de la imagen asociado al ID obtenido en el paso anterior:

   select id_imagen from palabra_imagen where id_palabra = 1;

   El resultado de efectuar la petición anterior se muestra en la Tabla 4.3.

id_palabra	**id_imagen**
1	2349
1	3565
1	6456
1	6712
1	13094
1	13317
1	28641
1	28642

Tabla 4.3: Asociación del id_palabra '1' con sus id_imagen.

   Al igual que ocurría en el paso anterior se selecciona, a priori, el primer ID de imagen resultado de la llamada anterior a la base de datos. Si no se obtiene pictograma con ese id_imagen es cuando se elige el siguiente, y así sucesivamente hasta la obtención de un pictograma o hasta terminar todos los IDs, caso en el que se volvería al paso 1 y se elegiría un nuevo id_palabra.

3. Con el id_imagen obtenido, se adquiere el pictograma alojado en la ruta correspondiente:

   http://dihana.cps.unizar.es/ pictograma/tablet/2349.png

   De esta forma, el pictograma asociado a la palabra *'comer'* se muestra en la Figura 4.1.

   NOTA 1: La última parte de la ruta está formada por el id_imagen junto con el formato de la imagen.

   NOTA 2: La penúltima parte de la ruta indica el tamaño de la imagen, pudiendo ser móvil o tablet. En este trabajo se utiliza el tamaño tablet.

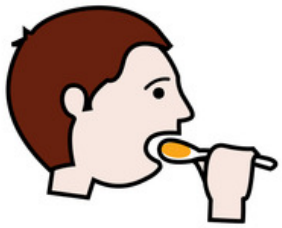


Figura 4.1: Pictograma asociado a la palabra 'comer'.

Para finalizar, el proceso explicado anteriormente se resume en la Figura 4.2.

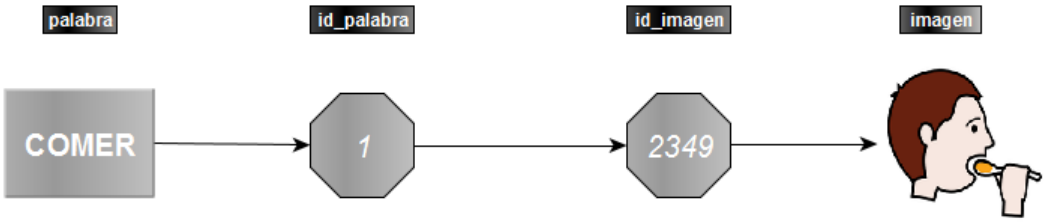


Figura 4.2: Resumen del proceso seguido para obtener un pictograma a partir de su palabra asociada.

4.1.1. Extensión de palabras con pictograma asociado.

El número de palabras que tienen al menos un pictograma asociado es de 4079 repartidas en 3514 pictogramas. Con el objetivo de ampliar esta población se llevan a cabo dos ideas. La primera de ellas es asociar todos las formas verbales al pictograma de su infinitivo, ya que sólo ellos tienen un pictograma asignado. La segunda idea se basa en la herramienta *Word2Vec* [35], que transforma cada palabra en un vector para asociar palabras por su parecido semántico.

Word2Vec toma como entrada un conjunto de textos y, utilizando redes neuronales, construye un contexto lingüístico de todas la palabras que contienen dichos textos. Cada palabras es representada en un espacio vectorial de varios cientos de dimensiones [35]. Así, cuando una palabra no tiene un pictograma asociado ni es una forma verbal, se le asocia el pictograma de una de las palabra que *Word2vec* ha identificado como similares. Si después de todo, no se consigue asociar un pictograma a una palabra se deja en blanco, dejando la posibilidad al usuario de introducirlo manualmente (ver sección 5.2.3.7). A continuación, se realiza un ejemplo de asignación de un pictograma a una palabra utilizando la herramienta *Word2Vec*.

Por ejemplo, la palabra inicial de la que no existe pictograma asociado en la base de datos es *'ordenadores'*. Al evaluar el modelo realizado con *Word2Vec* para obtener las 5 palabras más similares a *'ordenadores'* obtenemos el resultado mostrado en la Tabla 4.4. Este ejemplo ha sido realizado utilizando un modelo de lenguaje de gran cobertura, el cual ha sido creado a partir un gran cantidad de frases recogidas en noticias de televisión, plenos del Congreso de los Diputados y del Europarlamento, el tiempo, etc.

NOTA: El modelo de lenguaje utilizado en este trabajo para alimentar a la herramienta Word2vec es el descrito en la sección 4.2.

Palabra Inicial	Palabra Similar	Grado de Similitud
Ordenadores	Portátiles	0.682
Ordenadores	Archivos	0.677
Ordenadores	Ordenador	0.673
Ordenadores	Móviles	0.611
Ordenadores	Dispositivos	0.605

Tabla 4.4: Búsqueda de palabras similares a 'ordenadores' con la herramienta Word2Vec.

Una vez que obtenemos la lista de palabra similares evaluamos las palabras de una en una en orden descendente de similitud hasta que encontramos un pictograma asociado a alguna de las palabras. En este caso, *'portátiles'* y *'archivos'* tampoco tiene pictogramas asociado, pero sí *'ordenador'*. En la imagen 4.3 se puede ver el pictograma asignado a *'ordenadores'* gracias al uso de *Word2Vec*.

Figura 4.3: Pictograma asignado a la palabra 'ordenadores' a partir de la asociada a 'ordenador' mediante el uso de la herramienta Word2Vec.

4.2. Modelos de lenguaje.

Para crear el modelo de lenguaje global (ya que el caché se crea con la historia de cada usuario) sobre el que se asienta el algoritmo Viterbi de predicción explicado en el capítulo 3, se han utilizado cuentos populares y subtítulos de la cadena de televisión *Clan TV* dirigida a un público infantil. Estos subtítulos fueron recogidos entre octubre de 2015 y abril de 2016 y procesados eliminando las frases repetidas. Además, se añaden 3050 frases de uso cotidiano (compuestas por 22060 palabras).

En total, esta base de datos consta de 1795268 frases y 15708729 palabras. La división de las frases según su fuente se recoge en la tabla 4.5.

Fuente	Número de frases	Porcentaje	Número de palabras	Porcentaje
Cuentos populares	17120	**0.95** %	424293	**2.70** %
Clan TV	1775098	**98.88** %	15262376	**97.16** %
Uso Cotidiano	3050	**0.17** %	22060	**0.14** %
Total	**1795268**	**100** %	**15708729**	**100** %

Tabla 4.5: Desglose de la base de datos del modelo de lenguaje global utilizado en el algoritmo Viterbi según su fuente.

Por otro lado, las 3050 frases de uso cotidiano se utilizan también como sujetos de test para evaluar las prestaciones del sistema de predicción, dada su escasa relevancia en el modelo de lenguaje global, pudiéndose usar como sujetos de test independientes.

El modelo, como se ha explicado con anterioridad, se basa en N-gramas, llegando hasta los 5-gramas en el presente trabajo. Dicho modelo de N-gramas consta de:

- 4079 unigramas.
- 24366 bigramas.
- 25014 trigramas.
- 8184 tetragramas.
- 1436 cincogramas.

Además, como se explica en la sección 3.8, se utiliza otro modelo de lenguaje con todas las formas verbales y no sólo con infinitivos como es el caso del modelo anterior, para conjugar los verbos que incluye un frase una vez que ha sido predicha. Este modelo de lenguaje se crea con los mismo textos que el anterior. De esta forma, el modelo de lenguaje de N-gramas con conjugaciones lo conforman:

- 271570 unigramas.
- 122499 bigramas.
- 91404 trigramas.
- 31615 tetragramas.
- 7819 cincogramas.

De acuerdo a la cantidad de N-gramas que conforman ambos modelos, se pone de manifiesto la necesidad de haber hecho esta separación en base a agrupar toda la conjugación de un verbo en torno a su infinitivo, ya que el segundo modelo de lenguaje consta de un vocabulario 66.58 veces mayor que el primero (recordar que cada verbo contiene más de 100 formas verbales como se ha mencionado en la sección 3.8 pero que no todas aparecen en el corpus de textos utilizados. De ahí este dato). Con esta separación se gana en eficacia de la predicción a la vez que se conjugan los verbos.

Capítulo 5

TolkiChat

El producto final que se ha creado en este trabajo es el llamado *TolkiChat*. *TolkiChat* es una aplicación tipo chat que permite comunicarse a los usuarios mediante texto y pictogramas a la vez que genera predicciones conforme el usuario va escribiendo, actualizándose al utilizar la aplicación. Por tanto, se trata de un sistema de comunicación aumentativa y alternativa (CAA). Mencionar además que se trata de una comunicación broadcast, es decir, existe un único chat grupal al que acceden todos los usuarios que deciden utilizar la aplicación, comunicándose todos con todos.

En este capítulo se va a explicar, en primer lugar, la comunicación cliente-servidor sobre la que se asienta la aplicación *TolkiChat*. Posteriormente, se detalla la estructura de la misma, dividida en varias pantallas, así como cada una de las funciones de las que consta, como puede ser la muestra de los pictogramas ordenados por categorías o la predicción.

5.1. Comunicación cliente - servidor.

El sistema de chat está basado en una comunicación cliente - servidor. Todos los datos que un usuario envía son recibidos por el servidor, que puede devolver información al mismo usuario o reenviarlo al resto de clientes del sistema, según el tipo de mensaje del que se trate.

Los mensajes contienen una cabecera que hace posible la identificación del tipo de mensaje y una comunicación eficiente. Los diferentes tipos de mensajes son los siguientes: Inicio, Primer Inicio, Cambio Nickname, Imagen o Mensaje Chat. Así, cada tipo tiene su propia estructura de mensaje.

5.1.1. Inicio.

Cuando se comienza el programa, el cliente construye el siguiente mensaje para que el servidor le dé acceso al sistema:

[Inicio] + Nombre Usuario + Contraseña

Así, el servidor si el acceso es válido le responde dándole entrada y otorgándole sus datos de usuario:

[Inicio] + True + Número Referencia + Nombre Usuario + Nickname

En caso de que el acceso no sea válido el mensaje que devuelve el servidor es el siguiente:

[Inicio] + False + Motivo del rechazo

Los principales motivos de un acceso no válido son contraseña incorrecta o nombre de usuario inexistente.

5.1.2. Primer Inicio.

Cuando se accede al programa por primera vez, el usuario debe configurar su perfil, estableciendo un nombre de usuario, una contraseña y un nickname (opcional). El mensaje, por tanto, se compone como sigue:

[PrimerInicio] + Nombre Usuario + Contraseña + Nickname

Si se decide no asignar nickname, éste será el propio nombre de usuario:

[PrimerInicio] + Nombre Usuario + Contraseña + Nombre Usuario

La respuesta del servidor es la misma que ocurría en *Inicio*. El motivo del rechazo en esta ocasión es escoger un nombre de usuario que ya existe en la base de datos del sistema.

5.1.3. Cambio Nickname.

Cuando alguien decide cambiar de nickname, el mensaje que envía es el siguiente:

[CambioNick] + Nuevo Nickname + Nombre Usuario

En este caso, el servidor, simplemente, actualiza su base de datos.

5.1.4. Imagen.

Este mensaje se utiliza cuando el usuario decide personalizar o incluir un nuevo pictograma en su aplicación. Si un usuario utiliza los pictogramas por defecto, no tiene que enviarlos ya que al no notificar nada, el receptor contactará con la base de datos para mostrar esos mismos pictogramas. Si un usuario utiliza un pictograma personalizado, el emisor deberá enviar al receptor (vía servidor) todos los nuevos pictogramas. A su vez, el receptor es informado de en qué palabras debe utilizarlos (ver sección 5.1.5).

La imagen, al ser mayor que el tamaño máximo de cada mensaje, debe enviarse en diversos paquetes y, por tanto, son diferentes a los mensajes anteriores. A continuación se puede observar el contenido de un paquete de envío de imagen:

[Imagen] + Nombre Archivo + Dirección Usuario +

+ Datos Imagen + [(No)FinImagen] + Tamaño Datos Imagen

En el esquema de paquete anterior, el nombre de archivo especifica el formato de la imagen enviada (png, jpeg,...). También existe un marcador (*[FinImagen]*) para que el receptor sepa cuando ha recibido todos los datos del archivo para poder guardarlo. En caso de que no sea el último paquete se indica con *[NoFinImagen]*. Por último, se señala cuanto ocupan los datos relativos a la imagen en el paquete enviado a través del campo *Tamaño Datos Imagen*, al no ser un tamaño constante en cada paquete (el último tendrá, normalmente, menos bits de datos que los demás).

Con los datos proporcionados por el emisor, el receptor, al conocer esta estructura, podrá guardarse la imagen como pictograma y mostrarlo adjunto a su palabra asociada en el cuadro de conversación.

5.1.5. Mensaje Chat.

Finalmente, se describe la estructura que tiene un mensaje de chat, siendo éste el mensaje más utilizado en la aplicación. Esta es:

Dirección Usuario + Nickname + Mensaje + Información Pictogramas

El campo *Información Pictogramas* se trata de una cadena de 1's y 0's de longitud igual al número de palabras que contiene el mensaje. Así, se indica, en orden, con un 0 que esa palabra utiliza un pictograma por defecto y con un 1 un pictograma personalizado, procediendo posteriormente al envío o recepción, según el caso, de los pictogramas personalizados como ya se ha explicado.

5.2. Estructura de TolkiChat.

TolkiChat consta de tres pantallas principalmente, la pantalla *Inicio*, la pantalla *Nuevo Usuario* y la pantalla principal *TolkiChat*. Cada una de ellas, tiene diversas funcionalidades que se explican a continuación.

NOTA: En el Anexo B se pueden consultar los esquemas que resumen el funcionamiento de cada una de las pantallas que se describen a continuación.

5.2.1. Pantalla Inicio.

Esta pantalla da la bienvenida al usuario como se muestra en la figura 5.1. Se requiere que el usuario introduzca su nombre de usuario y su contraseña para posteriormente presionar la tecla *Confirmar*. Si el acceso es válido se hace click en la tecla *Continuar* para comenzar el chat. En caso contrario, se debe introducir o bien un nombre de usuario existente o bien la contraseña adecuada.

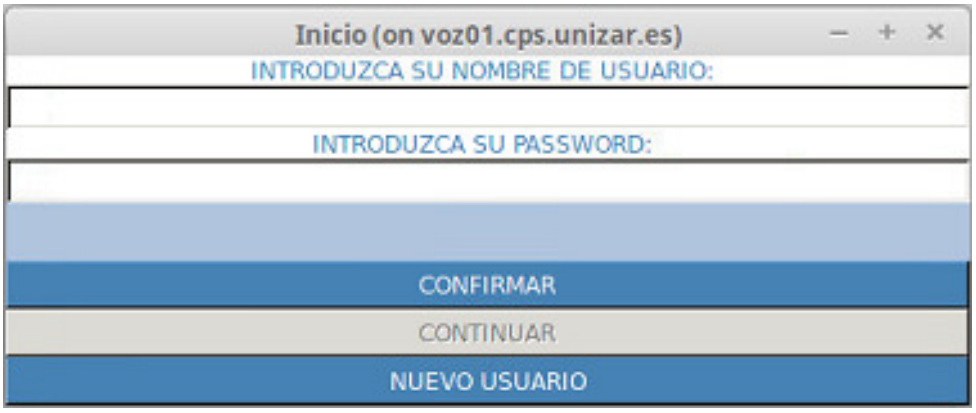


Figura 5.1: Pantalla Inicio de la aplicación TolkiChat.

Además, si es la primera vez que se inicia sesión se debe dar a la tecla *Nuevo Usuario* para acceder a la pantalla correspondiente para proceder a la creación de una cuenta, como se explica en la siguiente sección.

5.2.2. Pantalla Nuevo Usuario.

Esta pantalla posibilita la creación de un nuevo usuario, como su nombre indica. A través de la imagen 5.2, se puede observar que se debe introducir un nombre usuario (será el identificador del usuario en el sistema y, por tanto, debe ser único, no pudiendo coincidir con el de otro usuario) asociado a una contraseña. Además, se da la opción de introducir un nickname, que será el nombre con el que el resto de personas conectadas verán al usuario. Si no se introduce, el nickname por defecto será el propio nombre de usuario.

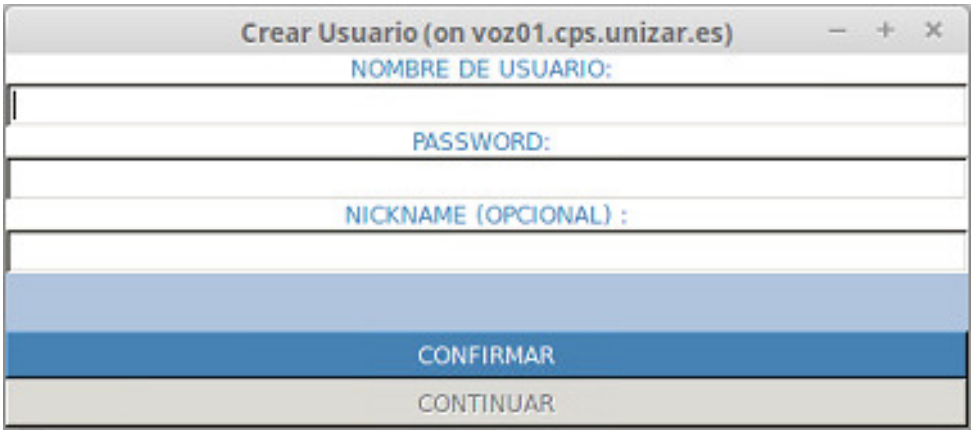


Figura 5.2: Pantalla Nuevo Usuario de la aplicación TolkiChat.

Como ocurría en la pantalla *Inicio* cuando se hayan introducido todos los datos se presiona el botón *Confirmar*. Si el acceso es válido (el único caso en el que no lo es, es al introducir un nombre de usuario ya existente) se pulsa la tecla *Continuar* para comenzar el chat.

5.2.3. Pantalla principal: Chat.

Una vez que se ha iniciado sesión a través de una de las pantallas anteriores, nos encontramos con la pantalla principal de la aplicación. Aquí se desarrolla toda la actividad de *TolkiChat*, es decir, el intercambio de mensajes basados en texto y pictogramas entre varios usuarios. Esta pantalla incorpora distintas funcionalidades que se explicarán a continuación.

En la Figura 5.3 se puede observar esta pantalla tras haber iniciado sesión.

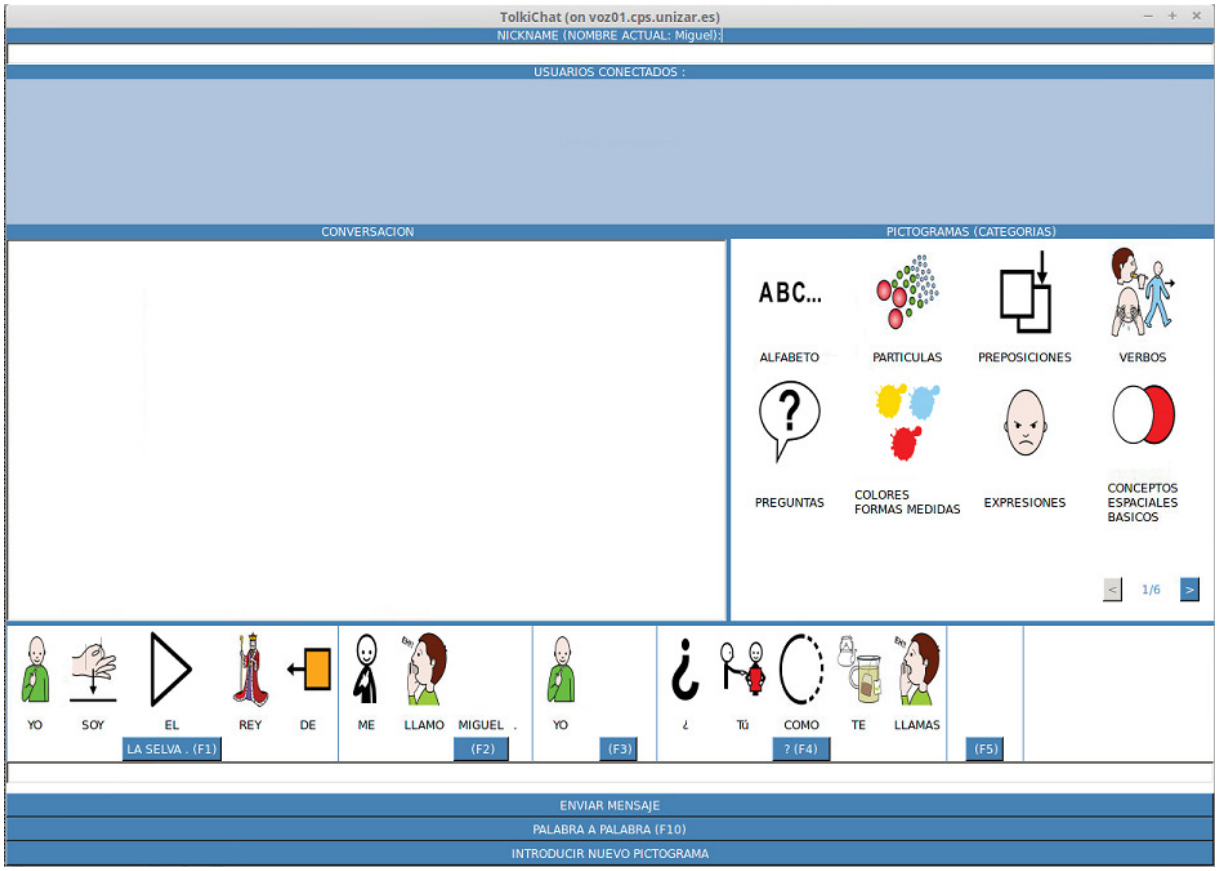


Figura 5.3: Pantalla principal (Chat) de la aplicación TolkiChat.

5.2.3.1. Cambio de nickname.

En esta apartado el usuario puede cambiar su nickname, es decir, el nombre con el que los demás le verán, cuando lo desee y tantas veces como quiera. Asimismo, entre paréntesis aparece el actual nickname del usuario.

En la imagen 5.4 se muestra la zona dentro de la pantalla principal donde se puede cambiar de nickname.

Figura 5.4: Cambio de nickname en la pantalla principal de TolkiChat.

5.2.3.2. Usuarios conectados.

En esta zona se muestran los usuarios que están conectados al chat. Como se aprecia en la Figura 5.5, la información de cada usuario se compone de un código que lo referencia y su nickname.

Figura 5.5: Indicación de los usuarios conectados en la pantalla principal de TolkiChat.

Inicialmente, el nickname que un usuario obtiene de los demás es el mismo código hasta que recibe un mensaje de cada usuario, momento en que actualiza su nickname al correspondiente. Esto es debido a que el servidor manda un mensaje automático a todos los usuarios en cuanto alguien se conecta, antes de que se identifique. Así, éste es añadido a la lista de usuarios conectados de todos los que ya se encuentran en el chat.

De la misma forma, cada vez que un usuario cambia de nickname, éste es notificado a todos con el primer mensaje suyo enviado al chat tras el cambio de nickname.

Finalmente, cuando alguien decide salir de la aplicación, es notificado a todos los usuarios y desaparece de la lista de conectados.

5.2.3.3. Conversación.

Esta es la zona más importante de una aplicación de chat, donde se muestra la conversación entre los diversos usuarios que están conectados.

En la Figura 5.6 se puede ver una conversación en *TolkiChat*. Se diferencia entre los mensaje enviados y los recibidos, presentándose estos últimos en color azul. Además los mensajes asociados a la conexión de un usuario se ofrecen en verde, mientras que los referentes a la desconexión se indican en rojo.

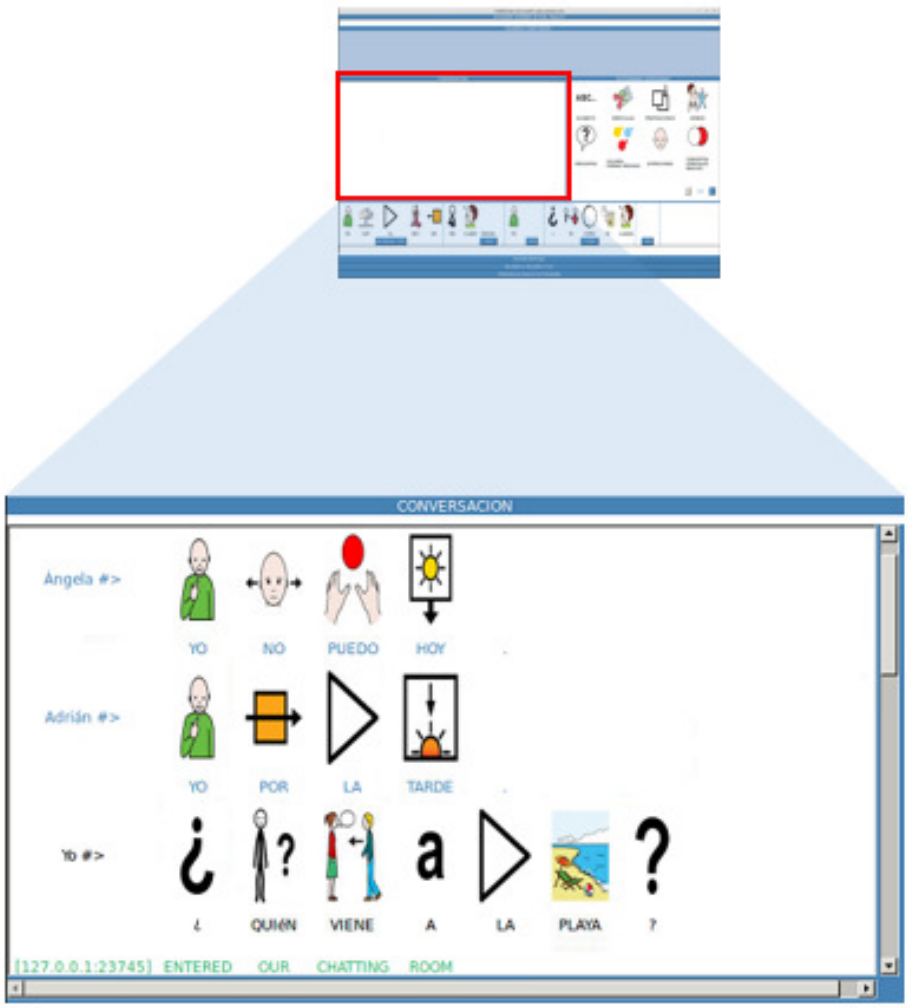


Figura 5.6: Muestra de una conversación en TolkiChat.

Conforme un nuevo mensaje llega, éste se presenta en la parte superior del cuadro de conversación, desplazando todos los demás mensajes hacia abajo. Se puede utilizar la barra de desplazamiento vertical para ver mensajes anteriores, y la horizontal para ver las últimas palabras de frases largas que no caben en el cuadro de conversación.

5.2.3.4. Pictogramas.

En este área aparecen los distintos grupos de pictogramas que existen. Como se observa en la parte inferior derecha de la Figura 5.7, se pueden utilizar los botones '<' y '>' para desplazarse a través de los distintos grupos de pictogramas.

Cabe destacar que estos grupos están ordenados por probabilidad de uso utilizando información de contexto, siendo actualizado su orden cada vez que el usuario interactúa con la aplicación, ya sea introduciendo una nueva palabra al cuadro de escritura (ver sección 5.2.3.6) o enviando una nueva frase. Así pues, en la Figura 5.7 aparecen los 8 grupos que son más probables que se utilicen al comenzar el chat.

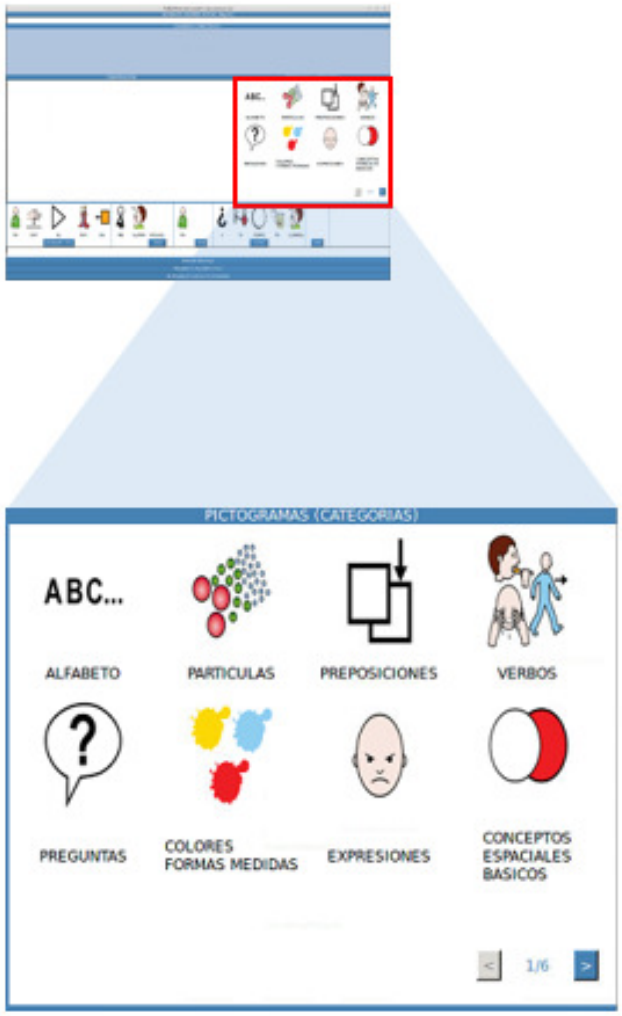


Figura 5.7: Grupos de pictogramas en TolkiChat.

Cuando se selecciona un grupo se accede a los pictogramas que éste contiene (como se puede ver en la Figura 5.8 al elegir el grupo *'Preguntas'*). De esta forma, cuando se selecciona ahora un pictograma, éste se incluye en el cuadro de escritura, así como su palabra asociada. Como ocurría cuando se mostraban los grupos, aquí la aparición de pictogramas también viene ordenada por su probabilidad de uso de mayor a menor. También se puede observar cómo se ha añadido el botón *'Volver a categorías'*, que como su propio nombre indica sirve para volver al menú de grupos.

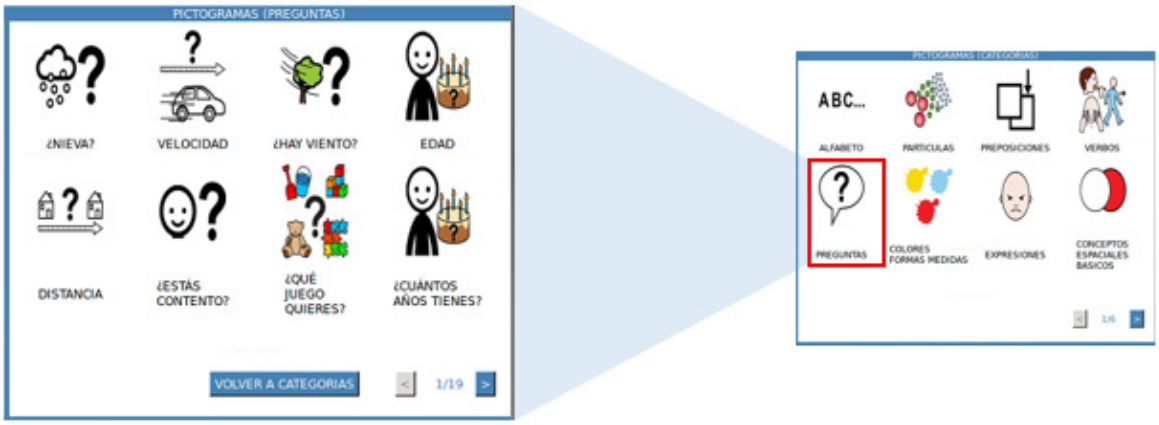


Figura 5.8: Grupo 'Preguntas' de pictogramas mostrados en TolkiChat.

5.2.3.5. Predicción.

En esta sección aparecen las predicciones que el sistema, basado en el algoritmo descrito en el capítulo 4, ofrece para ayudar al usuario a construir frases. Inicialmente, o cuando el usuario envía una nueva oración, el sistema presenta frases completas basado en la caché del usuario y el histórico de la conversación actual, tal y como se muestra en la Figura 5.9.

Por otro lado, cada vez que se escribe una nueva palabra el sistema propone las 5 opciones más probables para completar la oración, ahorrando así pulsaciones de teclado al usuario.

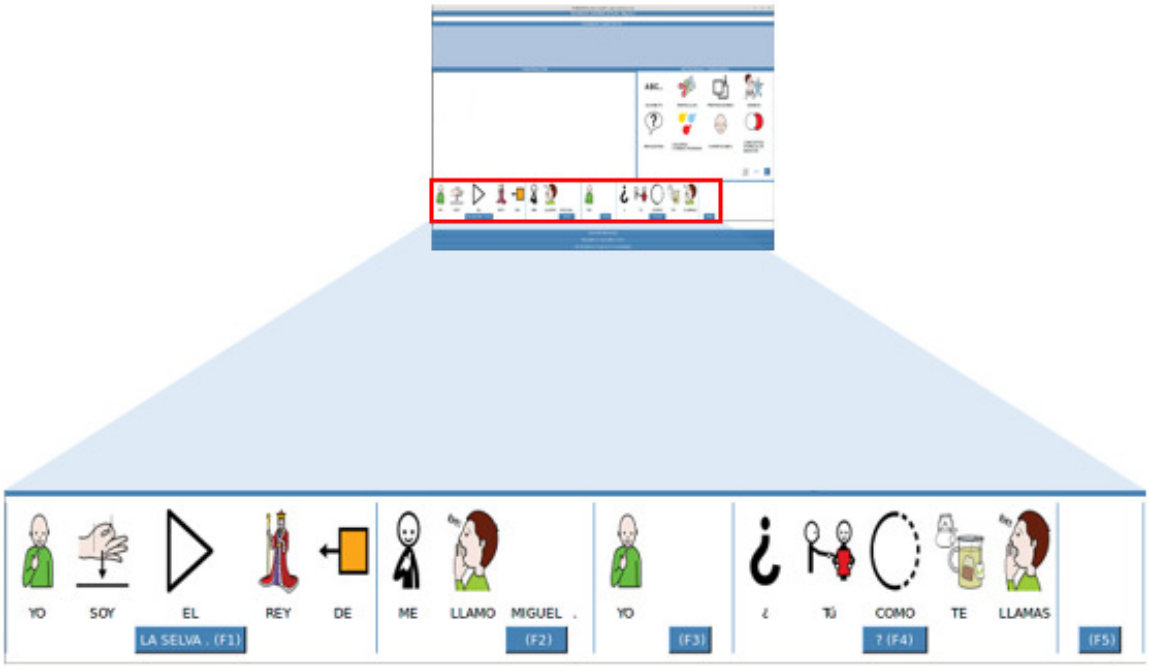


Figura 5.9: Parte de predicción en TolkiChat.

Para insertar la predicción preferida existen dos opciones, pulsando el botón azul correspondiente o bien utilizando el atajo de teclado indicado entre paréntesis.

Señalar que cada predicción está restringida a un máximo de 5 pictogramas; si la frase es más larga, las palabras restantes aparecen en el botón de selección de la predicción correspondiente junto con el atajo de teclado.

5.2.3.6. Escritura.

En esta parte el usuario puede ir escribiendo texto o, simplemente, ir seleccionando cada palabra que desee eligiendo su pictograma. En el primer caso, irá añadiéndose un nuevo pictograma cada vez que una nueva palabra sea escrita en la barra inferior como se muestra en la imagen 5.10. En el segundo caso, cuando se escoge un pictograma, se añade su palabra asociada al cuadro de texto.

En la Figura 5.10 también se observa la predicción actualizada de la frase que el usuario está escribiendo.

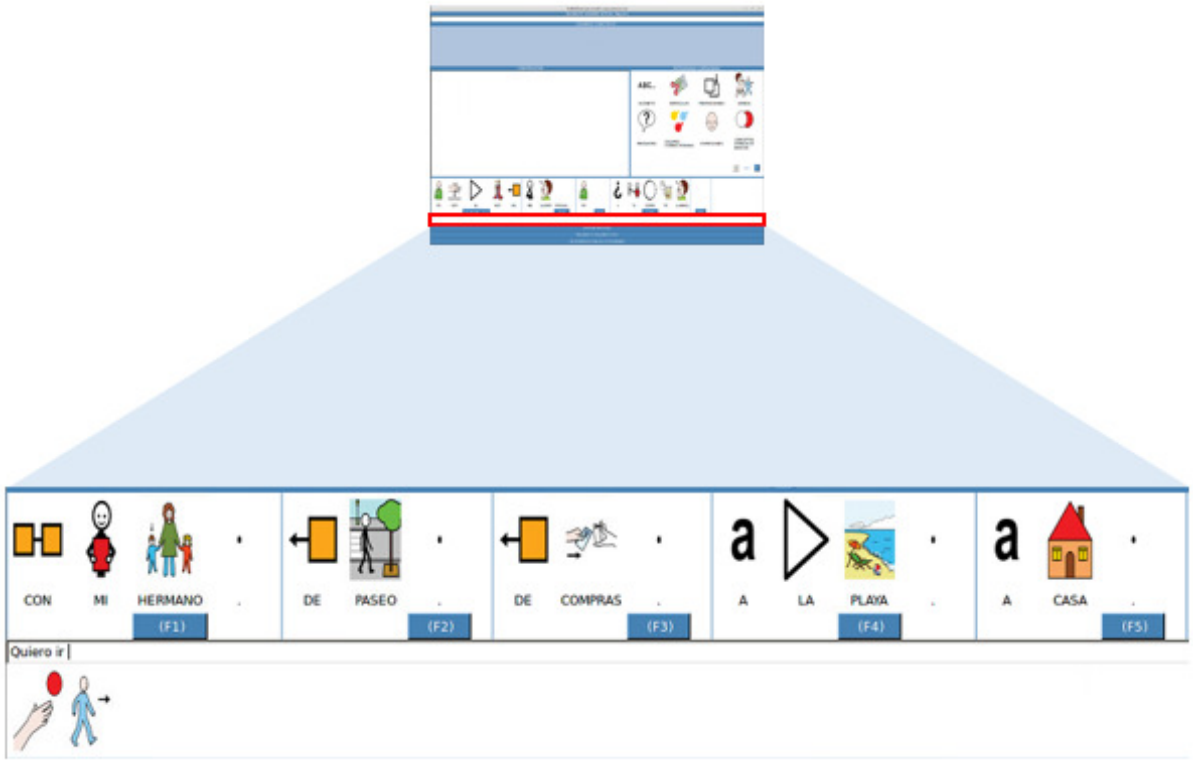


Figura 5.10: Escritura de una frase en TolkiChat y predicción asociada.

Cuando el usuario envía la frase, el sistema actualiza las distintas cachés: los modelos de lenguaje caché utilizado en la predicción (el que consta sólo de infinitivos y el que incluye todas la formas verbales), la caché de frases completas que se muestran cuando el cuadro de escritura aparece vacío y la caché de contexto utilizada para la ordenación de las categorías de pictogramas.

La caché de frases se realiza de forma equivalente a la caché utilizada para la predicción principal que ayuda al usuario a completar la frase que está escribiendo (ver sección 3.4). En vez de tomar palabras, esta vez cada elemento del N-grama lo componen la frase actual y las frases anteriores completas recibidas y/o enviadas por el propio usuario.

Sin embargo, para la ordenación de los pictogramas en la aplicación *TolkiChat* (ver sección 5.2.3.4), se utiliza información de contexto. Para ello, se traduce cada palabra a su contexto (a partir de la categorización hecha con sus pictogramas asociados, vista en la sección 4.1) construyendo un modelo de lenguaje caché de contexto, de la misma forma que se construyen los modelos de lenguaje caché anteriores (ver sección 3.4).

5.2.3.7. Botones inferiores.

Finalmente, en la parte inferior de la interfaz como se muestra en la Figura 5.3 aparecen tres botones. Con el primero de ellos se envían el texto y los pictogramas escritos; con el segundo se activa (o desactiva) el modo *'Palabra a Palabra'*. En este modo se puede ir introduciendo una predicción de una en una palabra. Esto es útil cuando haya una predicción acertada parcialmente, y no se desee introducir en su totalidad. Al igual que ocurre con las predicciones, para activar este modo se puede pulsar el botón correspondiente o utilizar el atajo de teclado *F10*.

El último botón sirve para introducir nuevos pictogramas, bien porque se quiera personalizar una palabra que ya tiene un pictograma asociado o bien porque una palabra (como puede ser un nombre propio) no disponga de pictograma. En la Figura 5.11 se aprecia la pantalla emergente que aparece al pulsar este botón. En ella, se requiere introducir una palabra y una imagen que haga de pictograma. Esta imagen se elige de entre las guardadas en el sistema del usuario mediante el botón *'Examinar...'*. Una vez se haya introducido la configuración deseada, esta se guarda mediante el botón *'Aceptar'*. Por otra parte, mediante el botón *'Limpiar'* se permite al usuario reiniciar la configuración por si ha cometido algún error.

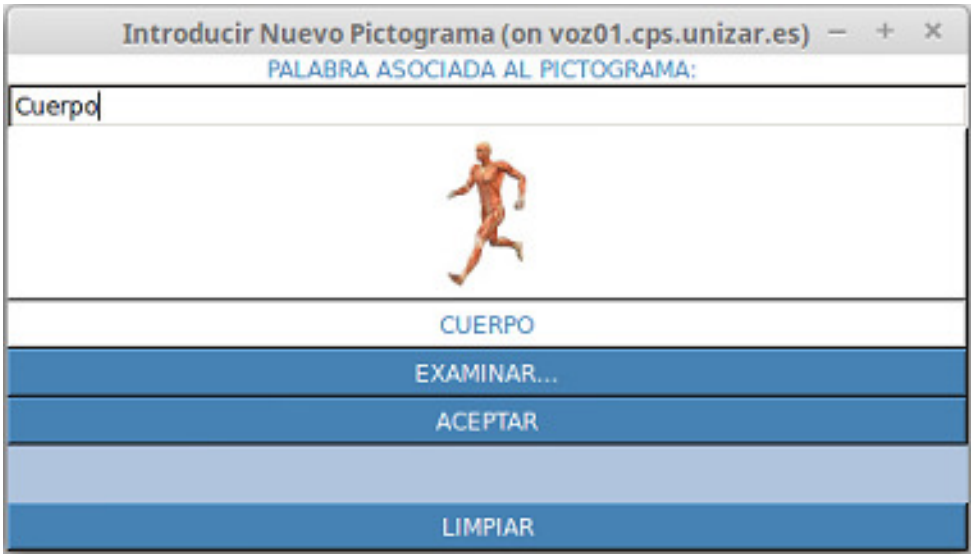


Figura 5.11: Pantalla de introducción de nuevos pictogramas en TolkiChat.

Capítulo 6

Resultados y discusión

En este capítulo se va a proceder a la evaluación del algoritmo Viterbi de predicción diseñado para este trabajo y explicado en el Capítulo 3. Como se mencionó en el Capítulo 4, las frases con las que se va a evaluar el sistema son 3050 frases de uso cotidiano, las cuales cuentan en total con 22060 palabras, esto es una media de 7.23 palabras por frase. Como la primera palabra de cada frase no es predecible, el total de palabras a predecir es de 19010.

Además, se van a ir discutiendo los resultados conforme vayan apareciendo.

6.1. Evaluación Inicial.

En primer lugar, se calcula la perplejidad según la Ecuación 3.7 vista en la sección 3.3 de las 3050 frases de test respecto al modelo de lenguaje global, así como respecto al modelo de lenguaje caché que va a ser generado. Estas medidas pueden observarse en la Tabla 6.1.

Modelo de Lenguaje	**Perplejidad**
Global	3332.99
Global con *UNK*	200.76
Caché	6.77

Tabla 6.1: Medidas de perplejidad para los distintos modelos de lenguaje.

La diferencia entre los modelos de lenguaje global y global con *UNK* es que en este último todas las palabras que no están en el vocabulario se agrupan en una sola (*UNK*), y, por tanto, se evalúan con la probabilidad de este elemento, mientras que si no se usa el *UNK* cada palabra se evalúa como una unidad fuera de vocabulario, aumentando la perplejidad del modelo.

El sistema se evalúa, en la mayor parte de experimentos, bajo cuatro condiciones referentes a la caché:

- **Sin Caché:** En este caso, la predicción se realiza teniendo en cuenta únicamente el modelo de lenguaje global.
- **Caché Progresiva:** Aquí la caché se va actualizando conforme se va evaluando cada frase.
- **Caché Inicial:** Se dispone de la caché de todas las frases que se van a evaluar desde el principio (resultado del proceso anterior). No se actualiza en este caso.
- **Caché Inicial + Progresiva:** Mezcla de los dos casos anteriores. Se dispone de la caché inicial y se va actualizando de nuevo tras evaluar cada frase.

En este primer experimento se van a evaluar las 3050 frases bajo las cuatro condiciones anteriores. El proceso de evaluación se lleva a cabo de la siguiente manera: se comienza con la primera palabra de cada frase como condición inicial. Si el algoritmo no acierta, se suma la segunda palabra a la condición inicial y se vuelve a solicitar una nueva predicción. Así sucesivamente hasta finalizar la frase (cuando se tengan más de N palabras como condiciones iniciales se tendrán en cuenta solo las N últimas -5 en este caso- para el modelo de N-gramas como se ha explicado en el Capítulo 3). En cambio, si el algoritmo acierta se suman todas las palabras acertadas a las condiciones iniciales para volver a realizar una nueva predicción.

Así, en la Tabla 6.2 se muestran el porcentaje de palabras acertadas, el número de palabras acertadas por pulsación, la precisión y el recall. El porcentaje de palabras acertadas se calcula como el cociente de todas las palabras acertadas en una frase entre la longitud total de la misma (exceptuando la primera palabra, la cual no se puede predecir al no tener condiciones iniciales). La ratio palabras acertadas por pulsación está enfocada a la aplicación *TolkiChat* y es el cociente entre el número de palabras predichas y el número de pulsaciones de los botones/atajos de teclado necesarias para introducirlas. La precisión se calcula al dividir el número de sugerencias realizadas por el sistema de predicción que son aceptadas entre el número de peticiones de predicción hechas al sistema (Ecuación 6.1), mientras que el recall es el cociente entre el número de sugerencias aceptadas y el número de veces que el sistema realiza al menos una sugerencia para completar la frase (Ecuación 6.2). Además, señalar que en los casos en los que la caché es utilizada, se utiliza $\alpha_{cache} = 0.50$ y $\alpha_{skip} = 0.20$ en la Ecuación 3.12.

$$precision = 100 \cdot \frac{\text{sugerencias aceptadas}}{\text{peticiones realizadas}} \quad (\%) \tag{6.1}$$

$$recall = 100 \cdot \frac{\text{sugerencias aceptadas}}{\text{sugerencias realizadas}} \quad (\%) \tag{6.2}$$

NOTA: Precisión y recall son calculadas en porcentaje.

	% **Pal. Acertadas**	**Pal. / Pul.**	**Precisión**	**Recall**
Sin Caché	9.15 %	1.10	8.37 %	14.85 %
Caché Progresiva	24.48 %	1.09	22.88 %	27.99 %
Caché Inicial	69.87 %	2.36	49.56 %	66.84 %
Caché Inicial + Progresiva	66.98 %	2.31	46.76 %	58.92 %

Tabla 6.2: Resultados de la evaluación de las 3050 frases de uso cotidiano bajo cuatro condiciones diferentes de utilización de la caché. Se muestran el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall.

Se observa la mejora de la predicción conforme el sistema va aprendiendo del uso, así como un aumento en el número de palabras introducidas por pulsación, lo que incrementa la eficiencia del sistema. La precisión y el recall también experimentan una mejoría conforme el sistema va aprendiendo. Sin embargo, una vez que la caché está disponible desde el principio, los resultados son mejores si el sistema no continúa aprendiendo.

Estos resultados están en la línea de las medidas de perplejidad ofrecidas en la Tabla 6.1. Con la diferencia entre la perplejidad del modelo global y el modelo global con *UNK* se observa la baja correlación entre el modelo de lenguaje global y las frases de test ya que se detectan una gran cantidad de palabras fuera de vocabulario. Además, se observa como la mejora de prestaciones está relacionada con la disminución de la preplejidad en el modelo de lenguaje caché respecto a los anteriores, lo cual hace más sencilla la tarea de predicción una vez que se dispone de su información. Finalmente, la diferencia entre las prestaciones del sistema en las evaluaciones *'Caché Inicial'* y *'Caché Inicial + Progresiva'* pone de manifiesto la baja correlación entre las propias frases de test (aunque mayor que entre las frases de test y el modelo de lenguaje global como se aprecia en la mejoría en el porcentaje de palabras acertadas entre

los casos *'Sin Caché'* y *'Caché Progresiva'*), siendo mejor disponer de la estadística total del conjunto de frases de test sin actualizar durante toda la evaluación que ir aprendiendo de nuevo frase a frase. Esto se confirma en la Figura 6.1, donde se analiza la perplejidad de cada frase respecto a cada una de los modelos de lenguaje caché en el momento en que va a ser evaluada. Se puede observar como la perplejidad aumenta conforme la caché es actualizada con más frases, siendo más difícil la tarea de predicción en la siguiente. Esto coincide con los datos de la Tabla 6.2, donde los resultados en *'Caché Inicial + Progresiva'* empeoran respecto a los de *'Caché Inicial'*.

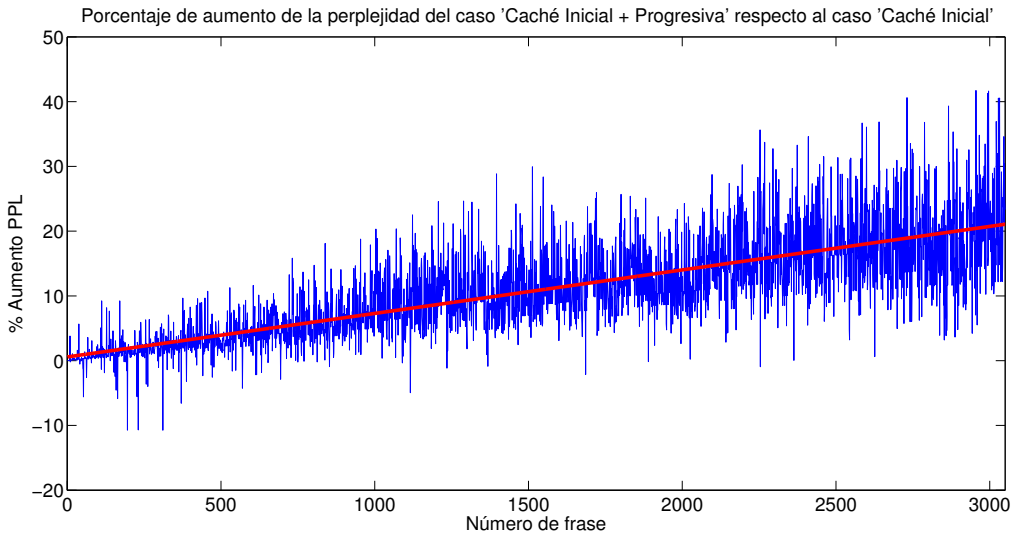


Figura 6.1: Evolución del porcentaje de aumento de la perplejidad del caso 'Caché Inicial + Progresiva' conforme se va actualizando el modelo de lenguaje caché respecto al caso 'Caché Inicial'. La línea roja es la recta de regresión que mejor se ajusta a los datos.

6.2. Evaluación de la Caché.

A la luz de los resultados anteriores se evalúa el sistema dando distintos pesos a la caché (Ecuación 3.12). En concreto se recorre α_{cache} entre 0 y 1 en pasos de 0.1 (en este caso, $\alpha_{skip} = 0.20$) . En la gráfica 6.2 se puede ver la evolución del porcentaje de palabras acertadas, mientras que en la gráfica 6.3 se muestra la progresión de la ratio palabras/pulsación.

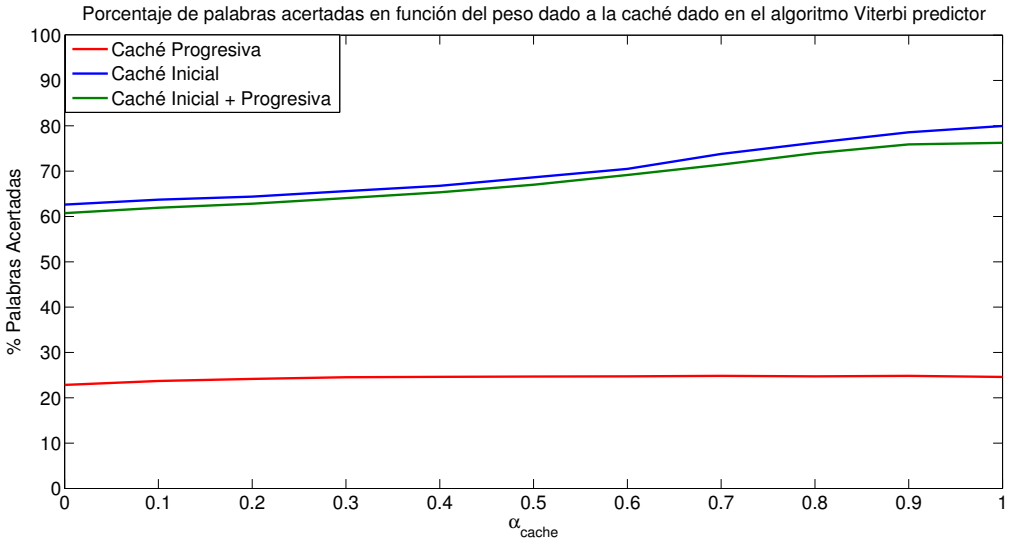


Figura 6.2: Evolución del porcentaje de palabras acertadas según el peso dado al modelo de lenguaje caché y al modelo de lenguaje global en el algoritmo Viterbi de predicción.

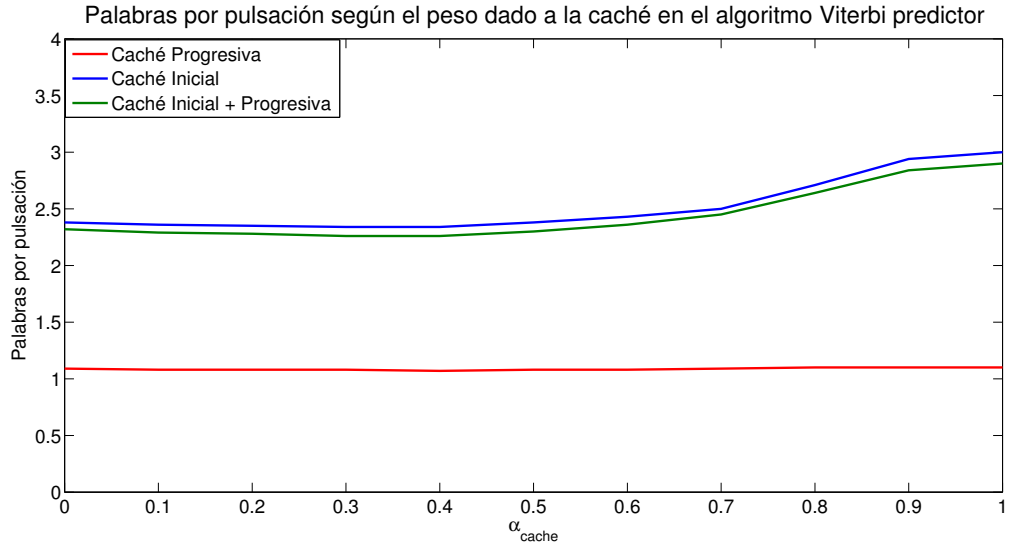


Figura 6.3: Evolución de la ratio palabras/pulsación de acuerdo al peso de la caché en el algoritmo Viterbi de predicción.

Las gráficas de las figuras 6.2 y 6.3 reflejan un mejor rendimiento conforme mayor peso se le da a la caché. En concreto, el número de palabras por pulsación presenta su mejoría a partir del 70 % de peso a la caché, llegando a las 3 palabras/pulsación cuando se da el 100 % del peso a la caché en el caso de *'Caché Inicial'*. En cuanto al porcentaje de acierto la subida es regular, subiendo del 70 % de acierto cuando se da más de un 60 % de peso a la caché y se dispone de ella desde el comienzo de la evaluación.

De la misma forma que la realizada para los parámetros anteriores, la evolución de la precisión y del recall se exponen en las gráficas 6.4 y 6.5 respectivamente.

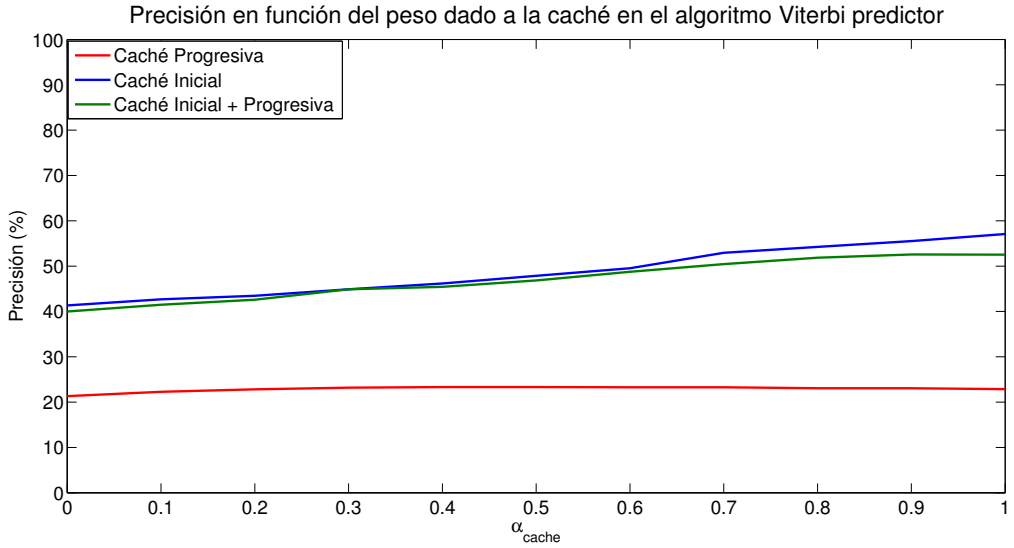


Figura 6.4: Evolución de la precisión de acuerdo al peso de la caché en el algoritmo Viterbi de predicción.

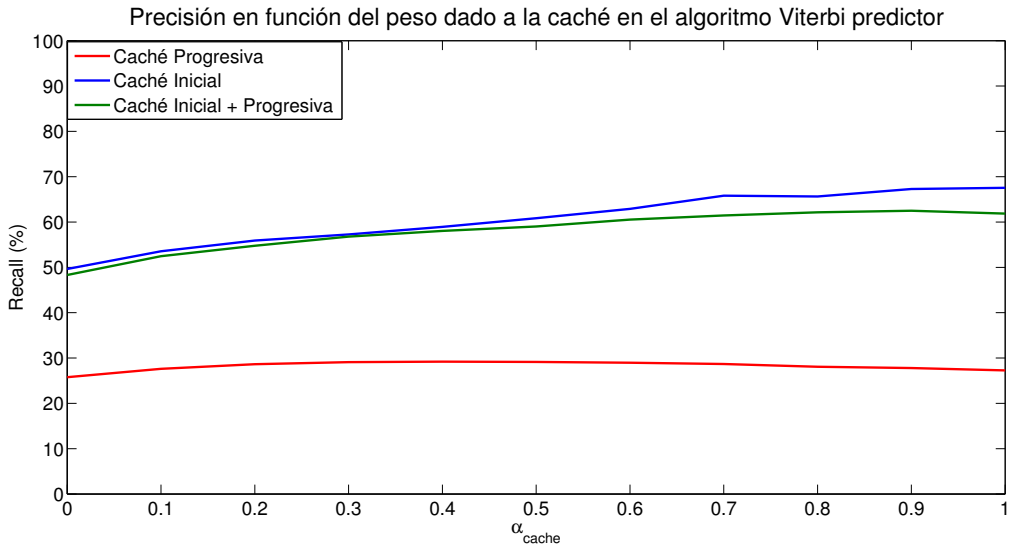


Figura 6.5: Evolución del recall de acuerdo al peso de la caché en el algoritmo Viterbi de predicción.

Como ocurre en el caso del porcentaje de palabras acertadas, la precisión y el recall mejoran conforme la importancia de la caché va aumentando en el algoritmo predictor. En concreto, la precisión supera el 50 % cuando se da un 70 % o más de peso a la caché y se dispone de ella desde el inicio de la evaluación. El recall supera el 60 % cuando el peso a la caché es más de 0.5 y, como en el caso de la precisión y el porcentaje de palabras acertadas, se utiliza desde el comienzo.

Las prestaciones son mucho mejores en los casos en los que ya se conoce la caché en su totalidad, en especial en el caso de caché inicial, ya que es donde se conoce mejor la estadística de todas la frases a analizar. Esto se ajusta a los resultados mostrados anteriormente, especialmente a las medidas de perplejidad donde la relativa al modelo de lenguaje caché es mucho mejor que la relativa al modelo de lenguaje global.

6.2.1. Evaluación de los Skip-gramas.

En este apartado, se valora la incidencia de los Skip-gramas en el algoritmo de predicción. En la Figura 6.6 se puede ver la evolución del porcentaje de acierto de la predicción en función de peso dado a los Skip-gramas en la Ecuación 3.12 (α_{cache} se fija a 0.5 durante la evaluación de los Skip-gramas). En el Anexo C se pueden consultar otros resultados de evaluar los Skip-gramas (ratio palabras/pulsación, precisión y recall).

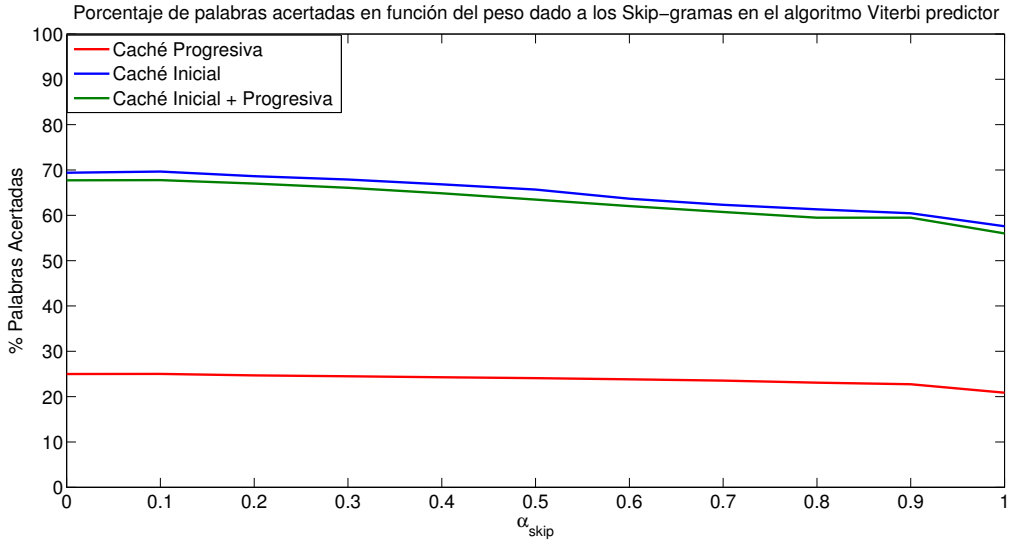


Figura 6.6: Evolución del porcentaje de palabras acertadas según el peso dado a los Skip-gramas el algoritmo Viterbi de predicción.

A la vista de la gráfica de la Figura 6.6, los mejores resultados se consiguen cuando $\alpha_{cache} = 0.10$. Esto indica que los Skip-gramas aportan información menos relevante que los N-gramas de la caché, pero más que el modelo de lenguaje global como se puede ver en las figuras 6.2, 6.3, 6.4 y 6.5, donde no utilizando caché pero sí Skip-gramas (datos en $\alpha_{cache} = 0$) los resultados son mejores que únicamente utilizando modelo de lenguaje global (caso *'Sin Caché'*).

6.3. Evaluación según la longitud de la frase.

A continuación, se evalúan las prestaciones del sistema en función de la longitud de la frase analizada en número de palabras. En la Figura 6.7 se muestra un histograma con el porcentaje de palabras acertadas para las distintas configuraciones de la caché (en el Anexo C se pueden consultar este tipo de histograma para la ratio palabras acertadas por pulsación, para la precisión y para el recall).

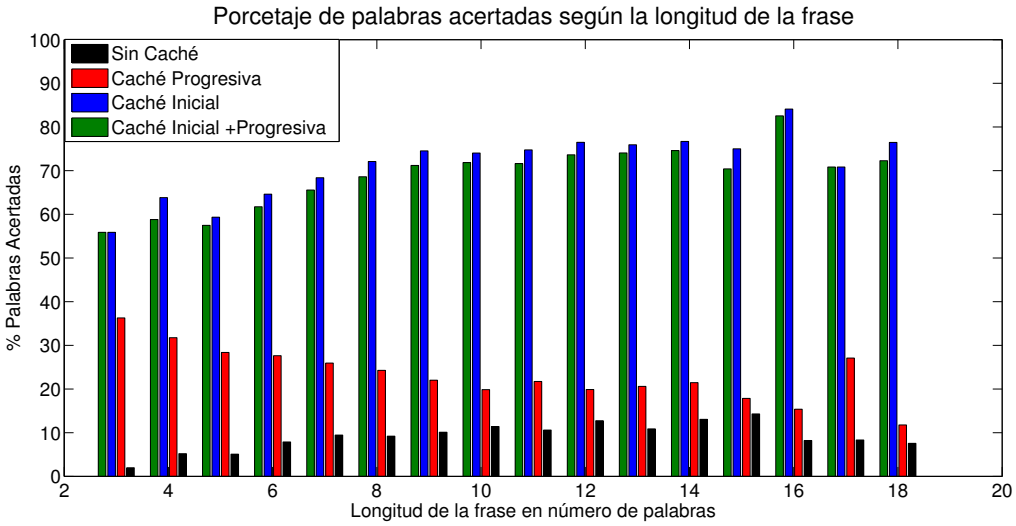


Figura 6.7: Porcentaje de palabras acertadas en función de la longitud de la frase.

En la Figura 6.7 se aprecia la notable mejoría, como se venía mostrando hasta ahora, del porcentaje de acierto cuando se dispone de la caché desde el comienzo de la evaluación. En cuanto a la longitud de la frase, las prestaciones del sistema aumentan cuanto más larga sea la frase, estabilizándose a partir de las 9 palabras.

6.4. Evaluación por grupos: Técnica LOOCV.

En esta sección dividimos el total de frases en 6 grupos aleatoriamente (cada grupo contiene una media de 508.33 frases) para evaluarlos con la técnica LOOCV (del inglés, *Leave One Out Cross Validation*), la cual busca evaluar cómo el sistema predictor funciona en un set de datos independiente [36]. Mediante esta técnica, se utilizan 5 grupos como sujetos de entrenamiento para construir el modelo de lenguaje caché, evaluándolo en el sexto de los grupos, utilizándolo como sujeto de test. Al haber dividido las frases en grupos, se busca evaluar cómo funciona la predicción en frases que no aparecen exactamente en la caché.

En la Tabla 6.3 viene expresada la media (y varianza) de evaluar diferentes parámetros en los 6 grupos mediante la técnica LOOCV. Los parámetros que se evalúan son el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall. En la fila de *'Train'* se presenta el resultado de evaluar todas las frases menos la del grupo en cuestión para aprender la caché. En la fila de *'Test'* se muestran los resultados de incorporar la caché obtenida del paso anterior y evaluar el algoritmo en el grupo de test (no incluido en la caché). Asimismo, en el caso *'Test Progresivo'* se ha evaluado de la misma forma que en el proceso de *'Test'* pero esta vez actualizando la caché cada vez que se valora una nueva frase. Finalmente, en el caso *'Test Total'* se evalúa la caché con el total de las 3050 frases, resultado del paso anterior (mismo caso que el de *'Caché Inicial'* en la Tabla 6.2).

La condiciones de la caché vuelven a ser $\alpha_{cache} = 0.50$ y $\alpha_{skip} = 0.20$ en la Ecuación 3.12.

	% **Pal. Ac.**	**Pal. / Pul.**	**Precisión**	**Recall**
Train	24.60 % (±0,03)	1.07 (±0,00)	23.33 % (±0,03)	28.94 % (±0,04)
Test	25.18 % (±0,52)	1.11 (±0,00)	23.34 % (±0,36)	29.85 % (±0,46)
Test Progresivo	25.58 % (±0,47)	1.11 (±0,00)	23.34 % (±0,38)	30.13 % (±0,57)
Test Total	69.39 % (±0,92)	2.38 (±0,00)	48.74 % (±0,86)	62.15 % (±0,70)

Tabla 6.3: Resultados en media (y varianza) de la evaluación mediante la técnica LOOCV para cada grupo de las 3050 frases de uso cotidiano (divididas en 6 grupos, 5 para entrenar la caché y el sexto como sujeto de test). Se muestran el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall.

Según los resultados de la Tabla 6.3, se observa como la gran mejoría viene en el caso *'Test Total'* cuando en la caché ya se han aprendido las frases que se van a evaluar. Por tanto, se observa que no hay datos suficientemente parecidos a las frases de test en la caché aprendida en la fase *'Train'* como para suponer un gran incremento de las prestaciones, siendo similares en los casos de *'Train'*, *'Test'* y *'Test Progresivo'*. Esto se confirma con el alto valor de la perplejidad de las frases de test con respecto al modelo de lenguaje caché realizado con las frases de entrenamiento. La media (y varianza) de evaluar la perplejidad para los 6 grupos es de 252.71 (± 66.39).

6.5. Evaluación TF-IDF.

A la vista de los resultados poco satisfactorios de la división por grupos anterior, realizada aleatoriamente, a continuación dividimos las frases por su semejanza.

En esta sección, se dividen las 3050 frases de test en grupos. Las frases se agrupan mediante la medida TF-IDF (del inglés, *Term frequency – Inverse document frequency*). Esta medida evalúa la relevancia de cada palabra en cada frase, compensada con la aparición de esa palabra en el corpus total de frases [37]. Una palabra tendrá una medida TF-IDF alta en aquellas frases en las que aparezca una gran cantidad de veces y no aparezca usualmente en el resto de frases. Así, utilizando esta medida se agrupan frases similares en 6 grupos, los cuales incluyen una media de 508.33 frases cada uno.

Además, cada grupo se divide en 5 subgrupos (101.77 frases de media incluidas en cada uno) para evaluarlos con la técnica LOOCV , como se ha realizado en la sección 6.4. Al haber dividido las frases en grupos mediante TF-IDF, se busca evaluar cómo funciona la predicción en frases que no aparecen como tal en la caché pero sí otras parecidas.

6.5.1. Modelo de lenguaje global: Total.

En este experimento, se utiliza como modelo de lenguaje global el mismo que se ha utilizado hasta ahora, el que incluye todas las frases de la base de datos (ver sección 4.2).

En la Tabla 6.4 se recogen los resultados en media (y varianza) de realizar la técnica LOOCV a cada uno de los 5 subgrupos de cada uno de los 6 grupos obtenidos mediante la técnica TF-IDF.

La columna de *'Train'* indica los resultados de ir adquiriendo la caché progresivamente al evaluar todas las frases de los 4 subgrupos de entrenamiento (todos menos el referente a la fila correspondiente). En la columna de *'Test'* se evalúan las frases del subgrupo restante con la caché adquirida en la fase de entrenamiento y sin actualizarla más durante todo este caso. Finalmente, en la columna de *'Test Progresivo'* se evalúan las frases de cada subgrupo partiendo del modelo de lenguaje caché adquirido en la fase de entrenamiento, pero además actualizando la caché cada vez que se evalúa una nueva frase.

	% Pal. Ac.	Pal. / Pul.	Precisión	Recall
Train	24.76 % (±3,53)	1.07 (±0,00)	23.52 % (±2,41)	26.97 % (±2,91)
Test	25.43 % (±9,31)	1.10 (±0,00)	24.01 % (±12,20)	29.13 % (±9,08)
Test Progresivo	25.96 % (±9,42)	1.12 (±0,00)	23.85 % (±5,66)	29.24 % (±8,23)

Tabla 6.4: Resultados en media (y varianza) de la evaluación con la técnica LOOCV de cada uno de los 5 subgrupos en todos los 6 grupos divididos mediante TF-IDF. Se muestran el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall.

6.5.2. Modelo de lenguaje global: Frases uso cotidiano.

En este caso el modelo de lenguaje global se cambia por las 3050 frases de uso cotidiano, buscando que se adapte mejor al contexto de las frases a evaluar.

Se valoran los mismos casos que en el apartado anterior. La Tabla 6.5 muestra los resultados de evaluar el mismo grupo que en la sección 6.5.1 tras las medidas TF-IDF y con la técnica LOOCV.

	% Pal. Ac.	Pal. / Pul.	Precisión	Recall
Train	39.44 % (±3,79)	1.24 (±0,00)	34.44 % (±2,71)	45.70 % (±2,05)
Test	40.91 % (±6,94)	1.31 (±0,01)	34.62 % (±6,48)	43.24 % (±6,72)
Test Progresivo	41.39 % (±13,70)	1.33 (±0,01)	34.66 % (±8,71)	42.92 % (±8,80)

Tabla 6.5: Resultados en media (y varianza) de la evaluación con la técnica LOOCV de cada uno de los 5 subgrupos en todos los 6 grupos divididos mediante TF-IDF, esta vez utilizando únicamente las 3050 frases de uso cotidiano como vocabulario global. Se muestran el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall.

6.5.3. Evaluación utilizando sólo caché.

A continuación se vuelve a analizar el mismo grupo pero esta vez dando todo el peso a la caché. Esta caché es la obtenida resultado de la fase *'Test Progresivo'* y no se actualiza frase a frase en esta evaluación. Resultados en la Tabla 6.6.

	% Pal. Ac.	Pal. / Pul.	Precisión	Recall
Solo Caché	84.42 % (±2,26)	3.57 (±0,38)	61.15 % (±5,08)	67.91 % (±7,53)

Tabla 6.6: Resultados en media (y varianza) de la evaluación con la técnica LOOCV de cada uno de los 5 subgrupos en todos los 6 grupos divididos mediante TF-IDF, esta vez utilizando únicamente la caché. Se muestran el porcentaje de palabras acertadas, la ratio palabras por pulsación, la precisión y el recall.

Finalmente, en la Tabla 6.7 se puede comprobar la mejora la división TF-IDF con respecto a la aleatoria, ya que la perplejidad de cada set de frases de test respecto a su modelo de lenguaje caché de entrenamiento es menor. Además, en la Tabla 6.8 se observan los datos de perplejidad en media (y varianza) de los conjuntos de frases de test respecto al modelo de lenguaje caché realizado con las 3050 frases de test inicial y con respecto a la caché realizada con su grupo TF-IDF correspondiente (las propias frases de test ya están incluidas en esta caché) .

NOTA: La desviación entre el dato de la Tabla 6.1 referente a la perplejidad de la caché y el dato de la Tabla 6.8 referente a la perplejidad de la caché total viene dada por la diferente dimensión de los grupos TF-IDF. Si se realizara una media ponderada de éste último dato, ambos serían iguales.

Modo de agrupar las frases	Perplejidad
Aleatoria	252.708 (±66.39)
TF-IDF	152.12 (±417.45)

Tabla 6.7: Perplejidad de las frases de test respecto a sus correspondientes frases de entrenamiento para las distintas formas de agrupar los datos.

Caché	Perplejidad
Total	6.58 (±0.91)
Grupo TF-IDF	4.43 (±0.04)

Tabla 6.8: Perplejidad de las frases de test respecto a las caché realizada con el total de las 3050 frases de test y con respecto a aquella únicamente realizada con su grupo de frases TF-IDF correspondiente.

Analizando las tablas 6.4 y 6.5 se manifiesta la importancia de adaptar el modelo de lenguaje global al contexto. Al utilizarse como modelo de lenguaje global aquel realizado con las frases de uso cotidiano (Tabla 6.5), se observa una mejoría de, aproximadamente, un 15 % en el porcentaje de palabras acertadas, un 10 % en la precisión y un 15 % en el recall respecto a utilizar el modelo de lenguaje global empleado en un principio. Esto se corrobora con el dato de la perplejidad, donde se baja de 3332.99 en el modelo global inicial a 6.77 en el modelo realizado con las frases de uso cotidiano (datos de la Tabla 6.1).

Por otro lado, en la Tabla 6.6 se observa una notable mejoría de los resultados al incluir en la caché las frases que posteriormente se van a analizar. Además, se obtienen los mejores resultados de entre todos los realizados. Esta mejora se basa, una vez más, en la mejora de la perplejidad. En este caso, el modelo caché utilizado se adapta mejor a las frases al haber utilizado la medida TF-IDF para agrupar las frases. En concreto, si comparamos los resultados de la Tabla 6.6 con los de la Tabla 6.3 pertenecientes al caso *'Test Total'* se experimenta una mejora de, aproximadamente, un 15 % en el porcentaje de palabras acertadas, un 13 % en la precisión y 5 % en el recall, así como un incremento de 1.19 palabras por pulsación. Esto viene dado por dos motivos: el primero de ellos es que en el caso de la Tabla 6.6 se utiliza únicamente la caché, mientras que en el de la Tabla 6.3 el peso de la caché es de únicamente el 50 % y, por lo tanto, el uso del modelo de lenguaje global hace que los resultados empeoren; el segundo de los motivos es que la caché se adapta mejor en el primer caso, como se observa en la Tabla 6.8, al tener una medidas de perplejidad de 4.43 (más de 2 puntos inferior a la relacionada con la caché total, utilizada en el segundo de los casos comparados).

6.6. Evaluación de verbos.

En esta sección se evalúa la tasa de acierto en la conjugación de verbos como recoge la Tabla 6.9, en la cual se indica el porcentaje de verbos acertados, de conjugaciones acertadas y el total acertado (es decir, conjunto de verbo y conjugación acertado). Se analiza para los casos sin uso de la caché, con actualización progresiva de la caché frase a frase y con disponibilidad inicial de la caché pero sin más actualización durante la evaluación. El modelo de lenguaje global vuelve a ser el construido con todas las frases de la base de datos.

	% Verbos Acertados	% Conj. Acertadas	% Acierto Total
Sin Caché	2.35 %	76.00 %	1.79 %
Caché Progresiva	9.86 %	61.23 %	6.04 %
Caché Inicial	43.19 %	83.55 %	36.09 %

Tabla 6.9: Resultado de analizar la conjugación de los verbos con distinto uso de la caché.

En la Tabla 6.9 se observan unos buenos datos de conjugaciones acertadas. Una vez más, el uso de la caché es importante a la hora de mejorar los datos como se aprecia, sobre todo, en el porcentaje de verbos acertados. Además, el porcentaje de conjugaciones acertadas en el caso *'Sin Caché'* es poco relevante al contar únicamente con 75 verbos acertados (en los casos *'Caché Progresiva'* y *'Caché Inicial'* se aciertan 315 y 1380 verbos respectivamente).

Finalmente en la Tabla 6.10 se muestran datos de perplejidad del modelo de lenguaje conjugado tanto introduciendo el elemento *UNK* como no, conforme se ha explicado en la sección 6.1.

Modelo de Lenguaje	**Perplejidad**
Global Conjugado	1678.86
Global Conjugado con *UNK*	160.76

Tabla 6.10: Medidas de perplejidad para los distintos modelos de lenguaje conjugados.

Los mejores resultados de la perplejidad de la tabla 6.10 respecto a la Tabla 6.1 son un indicio del aumento de conjugaciones acertadas respecto al de verbos. Pero sobre todo, esto es debido a que el número de posibilidades de conjugación de un verbo (alrededor de 100) es mucho menor que la longitud del vocabulario (más de 4000 palabras).

6.7. Comparativa con otros estudios similares.

Una vez que se han presentado los resultados de la evaluación del sistema de predicción realizados en este trabajo, se va a proceder a su comparación con otros estudios similares.

En el estudio realizado por *Weigand* y *Patel* [38] se consigue un 32 % de palabras acertadas con el sistema realizado a base de N-gramas y un 22 % de acierto con otro sistema realizado con Sem-gramas (N-gramas construidos a partir de información semántica). Este estudio utiliza alrededor de 140 millones de palabras escritas en blogs para construir tanto su modelo de lenguaje como sus frases de test. El 80 % de las frases totales son utilizadas para construir el modelo de lenguaje y el 20 % restante componen las frases de test.

De esta forma, los resultados anteriores son peores si se compara con el presente trabajo, donde introduciendo la caché junto con el modelo de lenguaje global, se llega al 69.87 % de acierto (Tabla 6.2) en la evaluación general, y al 84.42 % (Tabla 6.6) utilizando únicamente una caché mejor adaptada al contexto de las frases a evaluar. Si bien es cierto, que al no utilizar caché únicamente se alcanza un 9.15 % de palabras acertadas (Tabla 6.2). Una de las razones por la que los resultados de *Weigand* y *Patel* [38] son mejores que los del presente trabajo cuando no se utiliza la caché podría ser el mayor volumen de datos empleados para construir el modelo de lenguaje (en este trabajo se han empleado menos de 16 millones de palabras) y la aparente similitud entre las frases de test y modelo de lenguaje de entrenamiento.

Si se analiza el estudio de *Grabski* y *Scheffer* [20] basado en predecir el final de las frases a partir de frases con un inicio similar, se observa que obtienen, cuando la secuencia a predecir tiene más de 12 palabras, un 95 % de precisión y un 30 % de recall en el mejor de los casos. En este estudio [20], las frases utilizadas tanto en la fase de entrenamiento como en la fase de test han sido recogidas a partir de respuestas dadas vía correo electrónico en dos contextos distintos: consultas educativas y solicitudes en una tienda de ropa. En total, se disponen de 20.000 frases de las cuales, el 75 % se utilizan en el entrenamiento del modelo de lenguaje y el 25 % en el test. Esta división ha sido realizada aleatoriamente sobre el total de frases recogidas en los dos contextos señalados.

En el estudio de *Nandi* y *Jagadish* [19] se utilizan tres conjuntos de datos diferentes para construir el modelo de lenguaje: una conjunto de correos electrónicos enviados por la misma persona, una serie de correos electrónicos escritos por varias personas y una conjunto de datos extraídos de Wikipedia. El

total de los tres conjuntos están formados por, aproximadamente, 60.000 documentos. Las frases de test se obtienen del total de frases de los tres conjuntos anteriores.

Se obtiene, de media entre los tres casos analizados[19], un 84.78 % de precisión y un 23.99 % de recall cuando utilizan *FussyTree* explicado en el Capítulo 1 y una media de 88.93 % de precisión y un 36.99 % de recall cuando utilizan *FussyTree* reduciendo aquellos caminos de búsqueda que son poco significativos (no poco probables), según los criterios explicados en [19].

En el caso de este proyecto, si se consultan las tablas C.4 y C.5 del Anexo C, en las frases de más de 12 palabras se obtiene una precisión superior al 50 % y un recall en torno al 70 % para los casos de uso de caché total desde el comienzo de la evaluación. En general, la precisión obtenida en la evaluación inicial (Tabla 6.2) es de 49.56 % y el recall de un 66.84 % para el caso *'Caché Inicial'*. Por tanto, comparándolo con los estudios anteriores, en este proyecto se obtiene un mejor recall a costa de una precisión baja.

El bajo dato de la precisión en comparación con los estudios de *Weigand* y *Patel* [38] y de *Nandi* y *Jagadish* [19] puede deberse, una vez más, a la mejor adaptación de sus modelos de lenguaje a sus datos utilizados para el test, ya que son obtenidos en ambos casos de la misma fuente (o conjunto de fuentes). Esto no ocurre en el presente trabajo, como se ha mencionado en el Capítulo 4, ya que las frases que constituyen el modelo de lenguaje se han recogido de cuentos populares y subtítulos de *Clan TV* mayoritariamente, mientras que las frases de test entran dentro del contexto de uso cotidiano.

También es cierto que cuando en este trabajo se han utilizado los mismos conjuntos de frases para construir el modelo de lenguaje global y las frases de test (sección 6.5.2), la precisión se sitúa en torno al 35 %. La diferencia reside en que este modelo de lenguaje está formado por 3050 frases construyendo un modelo de lenguaje de poco tamaño en comparación con los estudios a los que se hace referencia [20] [19].

Capítulo 7

Conclusiones

El presente proyecto se enmarca dentro del campo de la comunicación aumentativa y alternativa, y tiene como objetivo el desarrollo de un sistema de diálogo que permita la comunicación mediante el uso de texto y pictogramas. A su vez, se incorpora un método de predicción que va dando sugerencias al usuario conforme va utilizando el sistema.

La predicción se incluye en tres partes del sistema de diálogo. La primera de ellas aparece al acceder a la pantalla principal de la aplicación, pues el sistema mostrará las frases más utilizadas por el usuario como sugerencia para iniciar la conversación. Asimismo, estas sugerencias irán apareciendo conforme se envíen o reciban mensajes. A su vez, la aplicación irá sugiriendo cómo se puede acabar una frase conforme el usuario está escribiendo, bien utilizando texto o pictogramas. Además, en la parte de selección de pictogramas también se usa el algoritmo de predicción. En primer lugar, las categorías en las que se dividen los distintos pictogramas (agrupación realizada gracias al uso de la base de datos de Arasaac) se ordenan utilizando información de contexto, mientras que los pictogramas dentro de cada categoría se ordenan según la probabilidad de que cada palabra ocupe la siguiente posición en la frase. Si aún no se ha comenzado la frase, simplemente aparecen las categorías ordenadas por relevancia y, dentro de cada una, las palabras se ordenan por probabilidad general de utilización.

Además, si una predicción no convence al usuario por completo pero sí parcialmente, existe la opción de activar el modo *'Palabra a Palabra'*. Con este modo se pueden ir introduciendo las palabras de cada predicción de una en una. Igualmente, la aplicación muestra qué usuarios están conectados al sistema en cada momento. También, existe la posibilidad de introducir nuevos pictogramas para aquellas palabras que no gozan de uno, o para personalizar palabras que ya tienen uno asignado.

Por otro lado, las prestaciones del algoritmo de predicción han sido evaluadas en el Capítulo 6. En él, se demuestra la importancia del uso de la caché, ya que hace incrementar notablemente el porcentaje de acierto de palabras, el número de palabras que se introducen cada vez que el usuario utiliza una sugerencia, la precisión y el recall. Esto se demuestra en los distintos experimentos y las gráficas que miden la evolución de estos parámetros en función del uso de la caché, donde cuanto más peso se le da, mejores resultados se obtienen. Asimismo, el uso de los Skip-gramas (los cuales se obtienen, como la caché, a partir del uso previo del sistema) también ha sido analizado y se concluye que cuando la caché es utilizada, no aportan una mejoría. En cambio, cuando la caché no es utilizada, el uso de los Skip-gramas permite una mejoría respecto al caso *'Sin Caché'* (observar gráficas de 6.2 a 6.5 con $\alpha_{cache} = 0$). Por tanto, cualquier información previa de utilización del sistema es mejor que la utilización del modelo de lenguaje global.

Así pues, se detecta una nula adaptación del modelo de lenguaje global a las frases evaluadas como muestran los datos de perplejidad de las frases evaluadas respecto a dicho modelo, y confirmado por los malos resultados que se obtienen cuando únicamente se utiliza este modelo y no la caché. Se demuestra la importancia de tener un modelo adaptado al contexto de la comunicación que se va a desarrollar. Cuando se cambia el modelo de lenguaje global por otro más adaptado al contexto de las frases evaluadas

(uso cotidiano en este caso), se consigue un aumento de, aproximadamente, un 15 % en el porcentaje de palabras acertadas, de un 10 % en la precisión y de un 15 % en el recall, así como de 0.2 en la ratio palabras/pulsación.

Siguiendo esta línea, se ha realizado también una evaluación LOOCV donde un grupo de frases se evalúa con la caché de otras. Las frases se han agrupado bajo dos puntos de vista, aleatoriamente y buscando similitud entre ellas mediante la técnica TF-IDF. Los resultados no arrojan una mejoría significativa, como cabría esperar dados los datos de perplejidad donde la relación de un conjunto de frases con su grupo de entrenamiento es 100 puntos inferior para el caso de agrupación mediante TF-IDF.

También se ha analizado el comportamiento del sistema según la longitud de la frase a predecir. De esta forma, se percibe una mejora del sistema para frases que no son cortas, esto es, a partir de las 8 palabras, especialmente en la ratio palabras/pulsación.

Los mejores resultados se obtienen cuando se usa únicamente la caché, consiguiéndose un porcentaje de acierto de 84.42 %, un 3.57 en el índice palabras/pulsación, una precisión de 61.15 % y un recall de 67.91 %.

Finalmente, si se comparan los resultados de este algoritmo con otros estudios similares, se observa que se obtienen unas mejores tasas de acierto en la mayoría de los casos, así como un mejor recall. El punto negativo viene dado por la precisión, donde los resultados son inferiores cuando se comparan con otros sistemas.

Capítulo 8

Líneas Futuras

Para finalizar esta memoria, se proponen una serie de líneas futuras con el propósito de mejorar tanto el algoritmo de predicción como el sistema de diálogo creado en este Trabajo Fin de Máster. El objetivo final de este proyecto debe ser la construcción de una aplicación que pueda ser utilizada por las personas con necesidades especiales, a las que va dirigida la comunicación aumentativa y alternativa.

Éstas son las siguientes:

- Conjugar los verbos teniendo en cuenta la información de las M palabras posteriores y no sólo de las N anteriores. De esta forma, podrán tenerse en cuenta más factores con el objetivo de realizar una mejor conjugación y mejorar la generación de lenguaje natural.
- Añadir información de contexto en el algoritmo de predicción. Así, se podrá ahorrar carga computacional y adaptar la predicción al contexto de la comunicación.
- Añadir información morfo-sintáctica en los modelos de lenguaje. De esta manera, se podrían unir palabras como se hace en el caso de los verbos, con plurales o géneros. Así, se podría mejorar la predicción al agrupar datos. Posteriormente, se debería adaptar el número y género de cada palabra, de manera similar a como se realiza en la conjugación de verbos.
- Mejorar los tiempos de predicción del algoritmo traduciendo la parte de acceso a los modelos de lenguaje de Python a C. Esta mejora permitiría mejorar definitivamente la fluidez del sistema.
- Permitir la comunicación además de con texto y pictogramas, mediante voz. Así, se añade un nuevo elemento más en este sistema de comunicación aumentativa y alternativa.

Acrónimos

% **Pal. Ac.**	Porcentaje de palabras acertadas
ARASAAC	Portal Aragonés de la Comunicación Aumentativa y alternativa
CAA / AAC	Comunicación Aumentativa y alternativa / Augmentative and Alternative Communication
LOOCV	Leave One Out Cross Validation (Validación Cruzada Dejando Uno Fuera)
NLP	Natural Language Processing (Procesado del Lenguaje Natural)
NLG	Natural Lenguage Generation (Generación del Lenguaje Natural)
Pal. / Pul.	Palabras por pulsación
POS	Part of Speech (Situación de la Palabra en el Texto)
TF-IDF	Term Frequency - Inverse Document Frequency (Frecuencia del Término - Frecuencia Inversa del Documento)

Bibliografía

[1] IRIS, "Towards natural interaction and communication." `http://www.iris-interaction.eu/`. Accedido 10-11-2016.

[2] E. de Discapacidad, "Autonomía personal y situaciones de dependencia (edad)," *Instituto Nacional de Estadística*, 2008.

[3] A. J. Lara and A. H. García, "Estadísticas y otros registros sobre discapacidad en España," *Política y sociedad*, vol. 47, no. 1, pp. 165–174, 2010.

[4] K. Trnka, D. Yarrington, K. McCoy, and C. Pennington, "Topic modeling in word prediction for aac," tech. rep., Technical report, 2005.

[5] J. L. Arnott and N. Alm, "Towards the improvement of augmentative and alternative communication through the modelling of conversation," *Computer Speech & Language*, vol. 27, no. 6, pp. 1194–1211, 2013.

[6] K. Gauda and M. Nowosad, "Computer support in non-verbal communication systems with using graphic sings in education of people with intellectual disabilities," *Advances in Science and Technology Research Journal*, vol. 8, no. 24, pp. 76–82, 2014.

[7] P. Mirenda, "Augmentative and alternative communication," *Handbook of Autism and Pervasive Developmental Disorders, Fourth Edition*, 1998.

[8] M. Okita, Y. Nakaura, and H. Suto, "A study of non-verbal communication system by using pictograms," in *Biometrics and Kansei Engineering, 2009. ICBAKE 2009. International Conference on*, pp. 205–208, IEEE, 2009.

[9] A. Waller and K. Jack, "A predictive blissymbolic to english translation system," in *Proceedings of the fifth international ACM conference on Assistive technologies*, pp. 186–191, ACM, 2002.

[10] Y. Netzer and M. Elhadad, "Using semantic authoring for blissymbols communication boards," in *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pp. 105–108, Association for Computational Linguistics, 2006.

[11] E. Dominowska, *A communication aid with context-aware vocabulary prediction*. PhD thesis, Massachusetts Institute of Technology, 2002.

[12] D. Park, S. Song, and D. Lee, "Smart phone-based context-aware augmentative and alternative communications system," *Journal of Central South University*, vol. 21, no. 9, pp. 3551–3558, 2014.

[13] S. Bickel, P. Haider, and T. Scheffer, "Predicting sentences using n-gram language models," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 193–200, Association for Computational Linguistics, 2005.

[14] N. Carmignani, "Predicting words and sentences using statistical models," *Departement of Computer Science, University of Pisa*, 2006.

[15] R. Kuhn and R. De Mori, "A cache-based natural language model for speech recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 6, pp. 570–583, 1990.

[16] N. Garay-Vitoria and J. Abascal, "Text prediction systems: a survey," *Universal Access in the Information Society*, vol. 4, no. 3, pp. 188–203, 2006.

[17] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine learning*, vol. 6, no. 1, pp. 37–66, 1991.

[18] T. Mikolov, "Statistical language models based on neural networks," tech. rep., Brno University of Technology. Faculty of Information Technology, 2012.

[19] A. Nandi and H. Jagadish, "Effective phrase prediction," in *Proceedings of the 33rd international conference on Very large data bases*, pp. 219–230, VLDB Endowment, 2007.

[20] K. Grabski and T. Scheffer, "Sentence completion," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 433–439, ACM, 2004.

[21] J. Light, "Toward a definition of communicative competence for individuals using augmentative and alternative communication systems," *Augmentative and Alternative Communication*, vol. 5, no. 2, pp. 137–144, 1989.

[22] Autismo Diario, "El lenguaje y la comunicación en niños con autismo." https://autismodiario.org/2016/08/29/el-lenguaje-y-la-comunicacion-en-ninos-con-autismo/, 2016. Accedido 21-10-2016.

[23] American Speech Language Hearing Association, "Augmentative and alternative communication (aac)." http://www.asha.org/public/speech/disorders/AAC/. Accedido 21-10-2016.

[24] M. Dempster, N. Alm, and E. Reiter, "Automatic generation of conversational utterances and narrative for augmentative and alternative communication: a prototype system," in *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*, pp. 10–18, Association for Computational Linguistics, 2010.

[25] A. Copestake, "Augmented and alternative nlp techniques for augmentative and alternative communication," in *Proceedings of the ACL workshop on Natural Language Processing for Communication Aids*, pp. 37–42, 1997.

[26] R. Dowse and M. Ehlers, "Medicine labels incorporating pictograms: do they influence understanding and adherence?," *Patient education and counseling*, vol. 58, no. 1, pp. 63–70, 2005.

[27] S. Davies, H. Haines, B. Norris, and J. R. Wilson, "Safety pictograms: are they getting the message across?," *Applied ergonomics*, vol. 29, no. 1, pp. 15–23, 1998.

[28] Baker, Bruce R, "Minspeak^{TM}History." http://www.minspeak.com/HistoryofMinspeak.php, 2009. Accedido 21-10-2016.

[29] VocalID^{TM}and FreeSpeech^{TM}, "Assistive technologies at the edge of language and speech science for children with communication disorders," 2015.

[30] V. C. Espín Leiva and A. B. Salazar Venegas, "Estudio de las funcionalidades y análisis comparativo del software *PiktoPlus*, utilizado para la comunicación aumentativa y alternativa, vs. la metodología tradicional no automatizada aplicada a personas con limitaciones de comunicación," 2015.

[31] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 179–190, 1983.

[32] S. Maskey, "Statistical methods for nlp. language models, graphical models.,"

[33] B. MacCartney, "NLP lunch tutorial: Smoothing," 2005.

[34] CAREI, Gobierno de Aragón, "Arasaac." http://carei.es/descipcion-arasaac/. Accedido 22-10-2016.

[35] Google, "Word2vec. tool for computing continuous distributed representations of words.." `https://code.google.com/archive/p/word2vec/`. Accedido 11-11-2016.

[36] Wikipedia, "Cross-validation (statistics)." `https://en.wikipedia.org/wiki/Cross-validation_(statistics)`. Accedido 13-11-2016.

[37] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, 2003.

[38] K. Wiegand and R. Patel, "Non-syntactic word prediction for aac," in *Proceedings of the Third Workshop on Speech and Language Processing for Assistive Technologies*, pp. 28–36, Association for Computational Linguistics, 2012.

ANEXOS

Anexo A

Cálculo del backoff mediante el método de Katz

Conforme se use el sistema se espera que el modelo de lenguaje de caché conste de un gran volumen de datos, superior incluso que el modelo global. Así, el parámetro de *backoff* en el modelo de lenguaje caché se ha calculado mediante el método de Katz [33]. Se ha elegido este método porque ofrece un mejor rendimiento cuando el modelo de *N-gramas* se entrena con una gran cantidad de datos que otros métodos como *Jelinek-Mercer* o *Kneser-Ney* (utilizado en el modelo de lenguaje global) [33].

Así, la probabilidad de *backoff* se ha calculado según la ecuación A.1.

$$P_{backoff}(\omega_{i-1}^{i}) = p_{katz}(\omega_i|\omega_{i-1}) = \frac{c_{katz}(\omega_{i-1}^{i})}{\sum_{\omega_i} c_{katz}(\omega_{i-1}^{i})} \tag{A.1}$$

Siendo $c_{katz}(\omega_{i-1}^{i})$:

$$c_{katz}(\omega_{i-1}^{i}) = \begin{cases} d_r r & \text{si } r > 0 \\ \alpha(\omega_{i-1}) p_{ML}(\omega_i) & \text{si } r = 0 \end{cases} \tag{A.2}$$

De la ecuación A.2, se deduce que r es el número de veces que ha aparecido el bigrama ω_{i-1}^{i}, mientras que d_r , $\alpha(\omega_{i-1})$ y $p_{ML}(\omega_i)$ se definen en las ecuaciones A.3, A.5 y A.6 respectivamente.

$$d_r = \frac{\frac{r*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \tag{A.3}$$

donde r*:

$$r* = (r+1)\frac{n_{r+1}}{n_r} \tag{A.4}$$

siendo n_r el número de n-gramas que han ocurrido r veces. Además, Katz sugiere el uso de k = 5 [33], y, por tanto, es lo que se ha utilizado.

$$\alpha(\omega_{i-1}) = \frac{1 - \sum_{\omega_i : c(\omega_{i-1}^{i})>0} p_{katz}(\omega_i|\omega_{i-1})}{1 - \sum_{\omega_i : c(\omega_{i-1}^{i})>0} p_{ML}(\omega_i)} \tag{A.5}$$

NOTA: $\alpha(\omega_{i-1})$ es elegida de acuerdo a que $c_{katz}(\omega_{i-1}^{i}) = c(\omega_{i-1}^{i})$.

$p_{ML}(\omega_i)$ es la probabilidad de máxima verosimilitud (del inglés, *Maximum Likelihood*). Por tanto:

$$p_{ML}(\omega_i) = \frac{p(\omega_i)}{\sum_{\omega_i} p(\omega_i)} \tag{A.6}$$

Anexo B

Esquemas de funcionamiento de TolkiChat

Este apéndice tiene como misión resumir el funcionamiento de todas las pantallas de la aplicación *TolkiChat* que se han explicado y mostrado en el Capítulo 5.

B.1. Inicio.

En la Figura B.1 se muestra el esquema de funcionamiento de la pantalla inicial, que da la bienvenida al usuario.

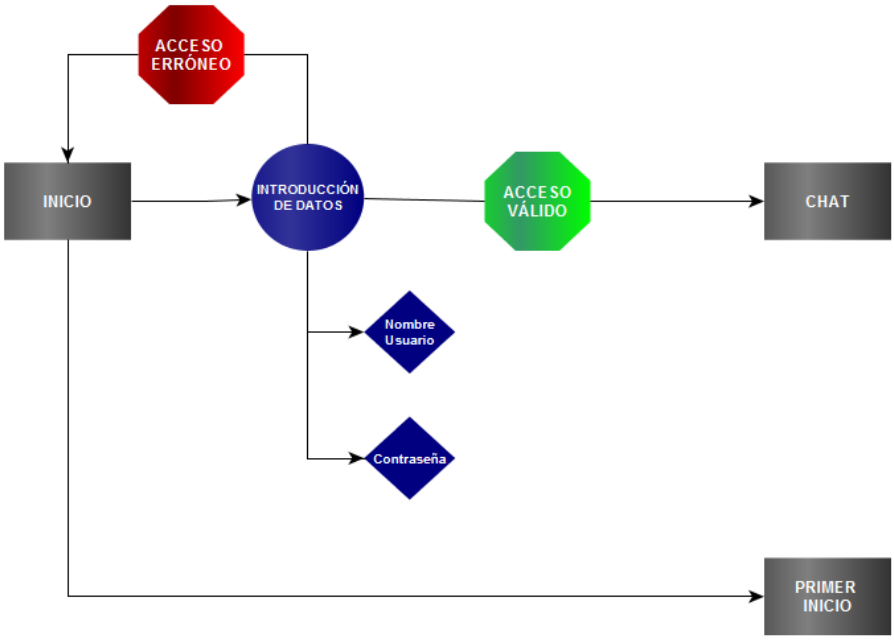


Figura B.1: Esquema de funcionamiento de la pantalla inicial de TolkiChat.

B.2. Primer Inicio.

El funcionamiento de la pantalla que permite el registro en el sistema cuando accedemos al programa por primera vez se esquematiza en la Figura B.2.

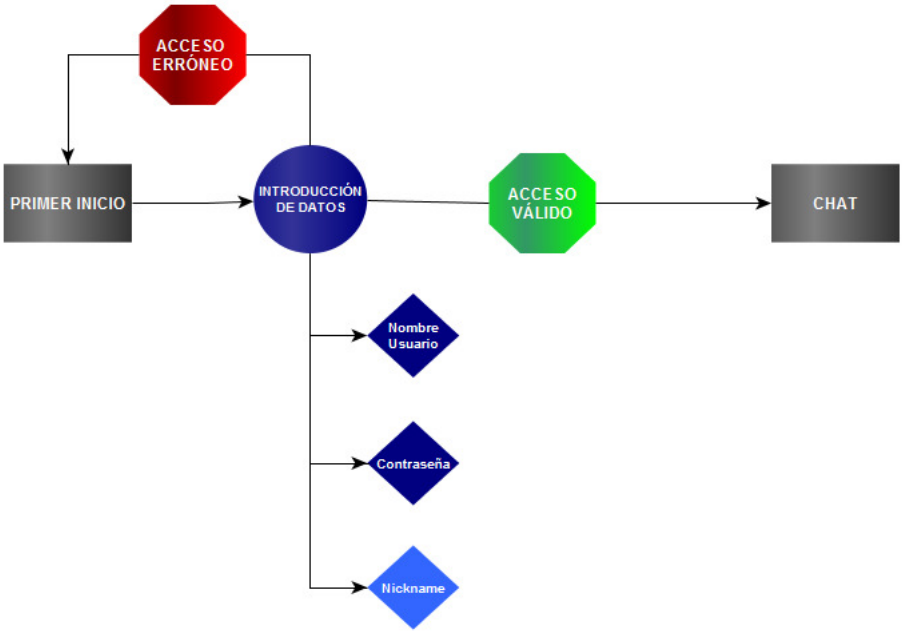


Figura B.2: Esquema de funcionamiento de la pantalla de registro de un usuario en el sistema de TolkiChat.

B.3. Chat.

En la Figura B.3 se describe el funcionamiento de todas las funcionalidades de la pantalla de chat, descritas en el Capítulo 5, de la aplicación *TolkiChat*.

NOTA: El cuadrado negro situado entre los octógonos 'Actualizar' y 'Usuarios Conectados' indica con un 1, que sólo se actualiza con el primer mensaje recibido por parte de cada usuario. En el resto de mensajes, ya se sabe que dicho usuario está conectado, y, por tanto, no procede actualizar la lista de usuarios conectados.

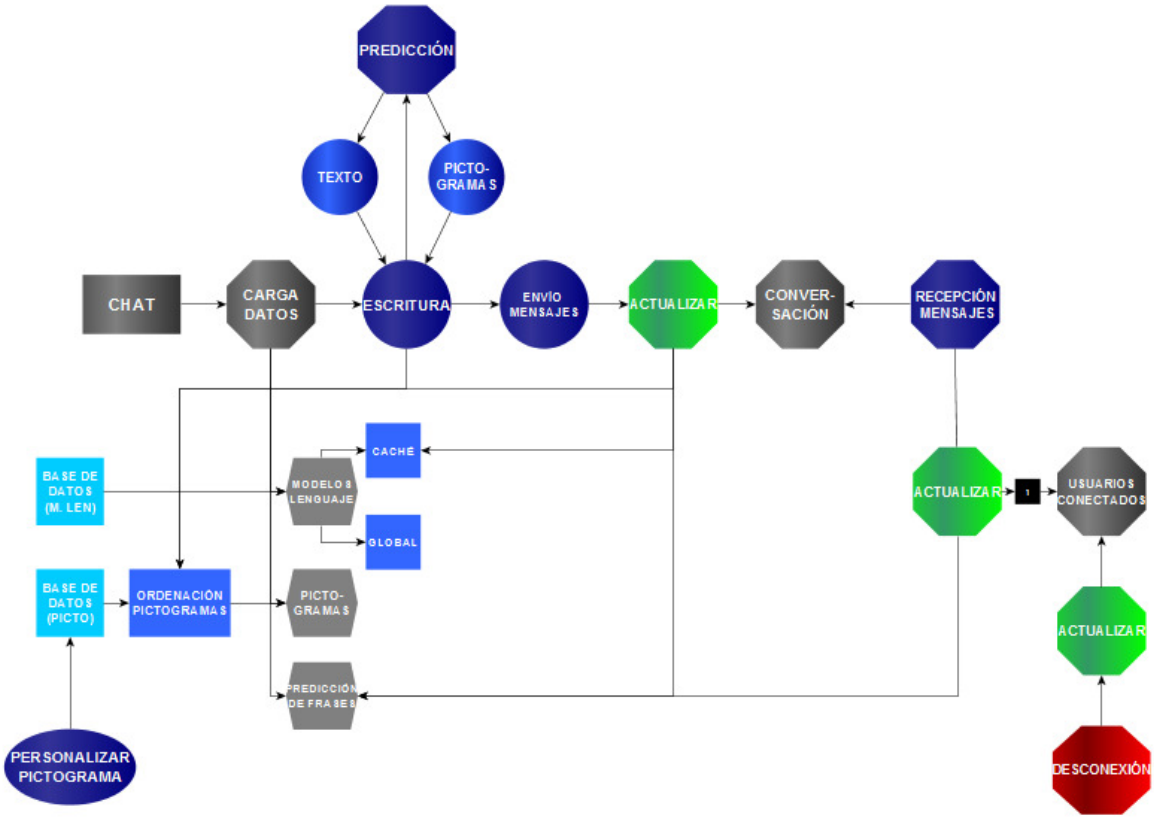


Figura B.3: Esquema de funcionamiento del sistema de chat en TolkiChat.

Anexo C

Extensión de resultados

En este anexo se muestran más resultados de la evaluación del algoritmo de predicción diseñado y que no aparecen en la memoria.

C.1. Evaluación de la caché.

C.1.1. Evaluación de los Skip-gramas.

En la Figura C.1 se muestra la evolución de la ratio palabras acertadas por pulsación en función del peso dado a los Skip-gramas para las distintas formas de utilización de la caché. Por otro lado, en la Figura C.2 se muestra como la precisión y el recall varían según la importancia que se le da a los Skip-gramas. Estos dos parámetros se han evaluado bajo el uso de caché inicial, y, además, una actualización de la misma cada vez que se evalúa una frase.

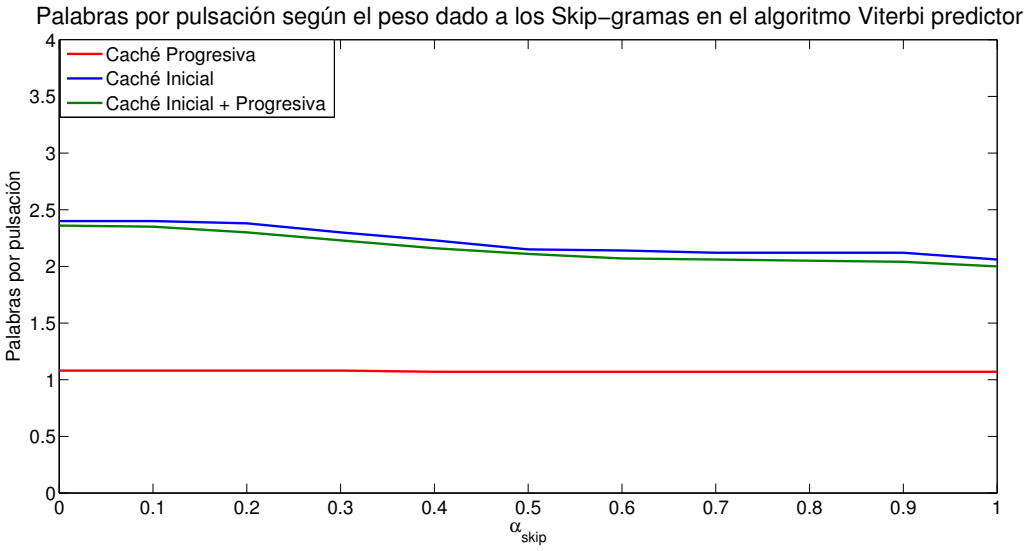


Figura C.1: Evolución de la ratio palabras acertadas / pulsación según el peso dado a los Skip-gramas el algoritmo Viterbi de predicción.

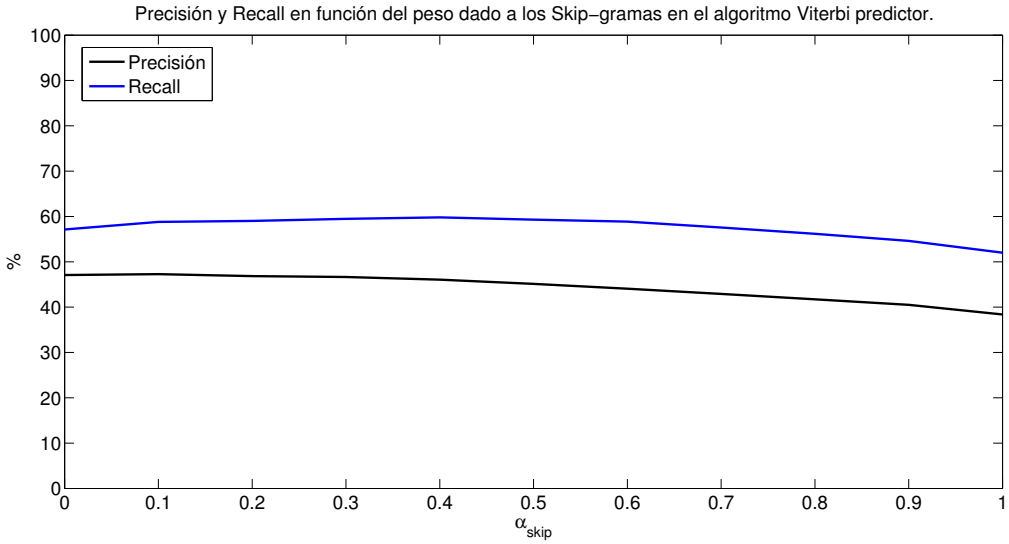


Figura C.2: Evolución de la precisión y el recall según el peso dado a los Skip-gramas el algoritmo Viterbi de predicción. Esta gráfica muestra los resultados en el caso 'Caché Inicial + Progresiva'.

Según los resultados mostrados en la Figura C.1, se registra una ligera disminución en la ratio *Palabras/Pulsación* conforme los Skip-gramas ganan importancia. Efecto similar ocurre en la precisión y el recall mostrados en la Figura C.2, siendo superior el recall a la precisión en más de un 10 %.

C.2. Evaluación según la longitud de la frase.

En esta sección se muestran más resultados del sistema de predicción en función de la longitud de la frase analizada en número de palabras. En la Figura C.3 se muestra el histograma referente a la ratio de palabras acertadas por pulsación, mientras que en las figuras C.4 y C.5 se exponen la precisión y el recall respectivamente.

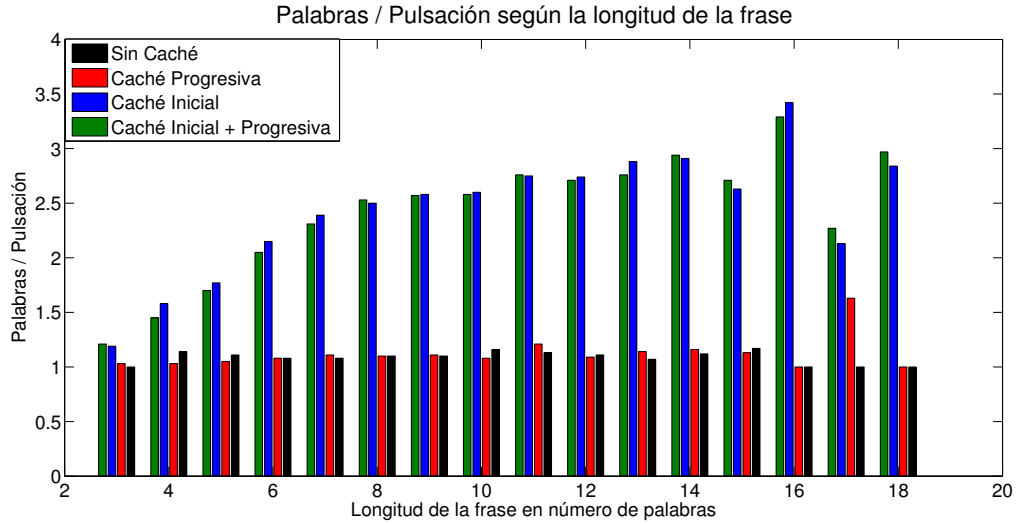


Figura C.3: Ratio palabras/pulsación en función de la longitud de la frase.

En la Figura C.3 se evidencia como el mayor ratio palabras/pulsación se encuentra en 14 palabras (si excluimos los datos en 16 y 18 palabras donde el número de datos es bajo y podrían no ser significativos) en los casos en los que se dispone de la caché total desde el comienzo (casos *'Caché Inicial'* y *'Caché Inicial + Progresiva'*)

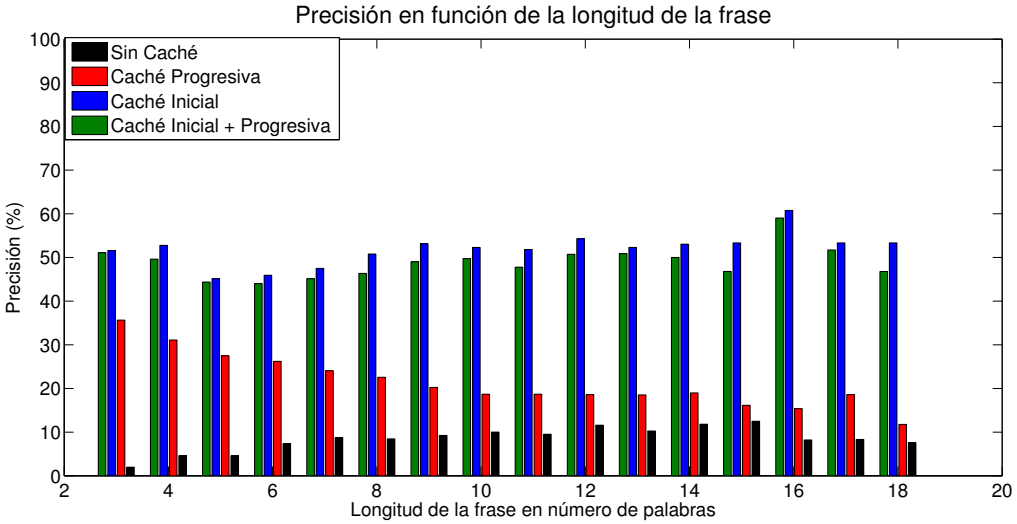


Figura C.4: Precisión en función de la longitud de la frase.

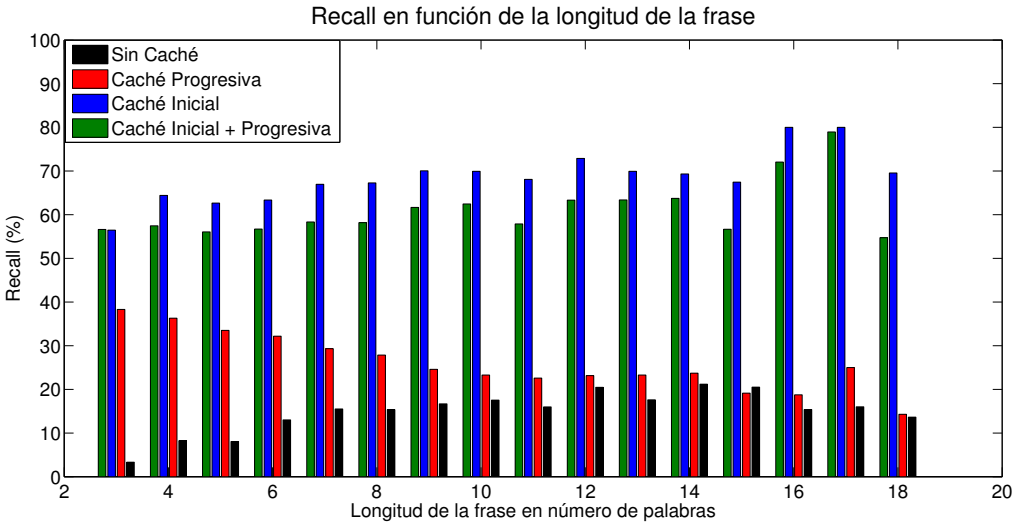


Figura C.5: Recall en función de la longitud de la frase.

En cuanto a la precisión y recall mostradas en las gráficas de las figuras C.4 y C.5 respectivamente, el mejor caso se encuentra en 12 palabras si excluimos las frases con más de 15 palabras donde los datos podrían no ser significativos. La precisión oscila entre el 50 % y el 60 %, mientras que el recall lo hace entre el 60 % y el 80 % en la mayoría de las ocasiones.