Universidad del País Vasco
Euskal Herriko Unibertsitatea

Máster Universitario
Ingeniería **C**omputacional y **S**istemas **I**nteligentes

Departamento de Ciencias de la Computación e Inteligencia Artificial

K I S A I C S I

Tesis de Máster

EMPATHIC-NLG: Un generador de lenguaje natural adaptado al coaching.

Alain Vázquez Risco

Directora

María Inés Torres Barañano

Departamento de Electricidad y Electrónica
Facultad de Ciencia y Tecnología (Campus Leioa)

Master eta Doktorego Eskola
Escuela de Máster y Doctorado
Master and Doctoral School

KZAA /CCIA

Septiembre 2019

Master eta Doktorego Eskola
Escuela de Máster y Doctorado
Master and Doctoral School

Índice general

Agradecimientos

En la elaboración de este trabajo han colaborado muchas personas. Por ello, no me quiero olvidar de dar las gracias tanto a mi directora como a mis compañeros de la Universidad del País Vasco. También agradecer las ayudas recibidas por parte de la empresa Acapela Group.

Capítulo 1

Introducción y objetivos del trabajo

El trabajo de fin de máster que se presenta a lo largo del siguiente texto se enmarca dentro del proyecto EMPATHIC[*], uno de los proyectos financiados por la Unión Europea dentro de lo que se conoce como Horizonte 2020[**]. La idea detrás de la financiación de estos proyectos en palabras de sus propios creadores es "apoyar las estrategias europeas en materia de Investigación, Desarrollo e Innovación Tecnológica, contribuyendo directamente a abordar los principales retos de la sociedad, así como la creación y mantenimiento de un liderazgo industrial europeo, y el refuerzo de la excelencia de la base científica". En este caso, se trata de un proyecto multidisciplinar en el que colaboran 10 instituciones de 6 países diferentes [1, 2, 3]:

- Universidad del País Vasco (coordinador)
- Osatek S.A.
- Oslo University Hospital
- e-Seniors Association
- Tunstall Healthcare (UK) Ltd.
- University of Barcelona
- Intelligent Voice Ltd.
- Acapela Group S.A.
- Institut Mines-Télécom
- Seconda Università degli Studi di Napoli

El reto H2020 para la sociedad en el que EMPATHIC se focaliza es el de mantener una sociedad de edad avanzada con un estilo de vida que les permita ser lo más independientes posible, lo que se conoce como envejecimiento saludable. Dicho reto surge ante el problema desde las instituciones sanitarias para poder definir un sistema de cuidados para personas de edad avanzada que cubra las necesidades de la sociedad. Actualmente, dichas carencias normalmente son cubiertas con los cuidados llevados a cabo por familiares o amigos. Sin embargo, las predicciones dicen que cada vez la disponibilidad de dichos cuidados se va a ver reducida. Es por ello que los esfuerzos están puestos en incrementar el número de personas mayores independientes. Algunos estudios establecen que el foco para poder conseguir un envejecimiento saludable debe estar puesto en el

[*] http://www.empathic-project.eu/
[**] https://www.horizon2020.es/

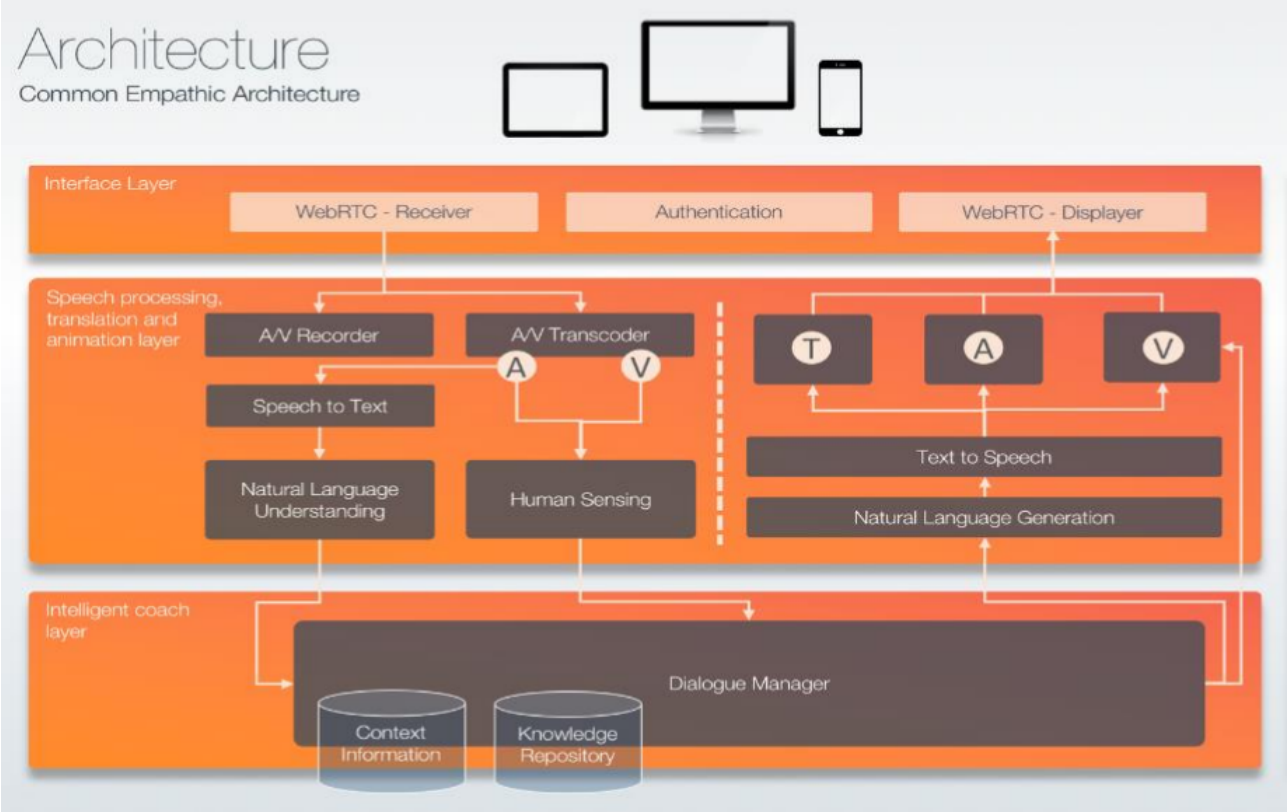

Figura 1.1: Arquitectura del EMPATHIC-VC.

estilo de vida de las personas mayores [4, 5]. Así, la idea sería promover un estilo de vida activo y saludable teniendo en cuenta las dificultades habituales que puedan encontrar las personas mayores de 65 años, que es la población sobre la que actúa EMPATHIC.

Apoyado en la idea de que una posible solución podría estar en el uso de robots o asistentes virtuales [6, 7], EMPATHIC busca diseñar su propio *virtual coach* (VC) con el que investigar, innovar y validar las bases para la creación de la nueva generación de VC personalizados con el que poder mantener a los mayores dentro de la sociedad activa. En este sentido, el EMPATHIC VC es un sistema remoto basado en tecnologías no intrusivas con capacidad para extraer marcadores fisiológicos de estados emocionales en tiempo real dando una capacidad empática al VC. Las otras dos características asociadas a la tecnología usada son la preservación de la privacidad y la expresividad del coach.

Por otro lado, el proyecto está involucrando usuarios reales desde su inicio ya que la idea es validar su efectividad y eficiencia con ellos. Dichos usuarios finales son personas de más de 65 años de tres sociedades europeas distintas: España, Francia y Noruega. Es por ello, que el VC va a estar desarrollado en las tres respectivas lenguas oficiales de dichos países: castellano, francés y noruego. Por último, la idea es obtener modelos de interacción entre coach y usuario adaptables a cada uno de ellos pero manteniendo ciertas pautas de lo que sería una interacción real con un coach profesional.

El EMPATHIC-VC es un Sistema de Diálogo Hablado (Spoken Dialog System, SDS) ya que se trata de un sistema que cumple la característica de proporcionar una interfaz al servicio a través del diálogo hablado, entendiendo diálogo hablado como una interacción maquina-humano de más de un turno con uso del lenguaje natural mediante voz. Dichos sistemas presentan una arquitectura (figura 1.1) que implica a varios módulos con distintas funciones: reconocer lo que

ha dicho el usuario, extraer la información relevante de lo dicho, decidir una respuesta adecuada, generar el texto asociada a ella y sintetizar la voz que proporcione la respuesta al usuario. Para captar tanto las entradas del usuario como para poder generar las salidas con formato audiovisual se aprovechan las funcionalidades de los ordenadores, móviles o tablets donde se tiene intención integrar el sistema desarrollado en EMPATHIC.

De todos los módulos explicados en el anterior párrafo, este trabajo va a versar sobre el módulo encargado de generar el texto de respuesta del sistema. Dicho componente se denomina Generador de Lenguaje Natural (Natural Lenguage Generator, NLG). Como parte de dicha arquitectura, se va a ver que el generador descrito a continuación está muy adaptado a las necesidades del proyecto. Por un lado, se podrá ver que los esfuerzos han ido encaminados a cubrir las posibles entradas que el gestor de diálogo (módulo previo al NLG) pueda generar y adaptar la salida a un formato útil para que la síntesis a voz del texto se haga del modo más correcto posible. Otro de los retos ha sido el desarrollo un NLG multilingüe, es decir, que el generador tiene que trabajar correctamente en castellano, francés y noruego, siendo estos dos últimos idiomas nada familiares para el autor. A lo largo del trabajo se explica cómo se ha lidiado con dicha problemática.

Para una comprensión más sencilla de los procesos llevados a cabo durante este trabajo el escrito se ha dividido de la siguiente manera: el capítulo 2 describe el estado del arte del NLG, donde se trata el tema de un modo general. Seguidamente, en el capítulo 3 se detalla el proceso experimental seguido para el desarrollo del NLG-EMPATHIC, que implica la creación de una base de datos adaptada a las necesidades de EMPATHIC y la construcción de dos tipos de generadores. En el capítulo 4, se detallan los resultados estadísticos del etiquetado de la base de datos y se realiza una evaluación de los generadores. Finalmente, el capítulo 5 cierra el trabajo con una serie de conclusiones e ideas para la continuación en la línea de investigación del NLG-EMPATHIC.

Capítulo 2

Generación de Lenguaje Natural: estado del arte

La Generación de Lenguaje Natural (Natural Language Generation, NLG) es un subcampo de la inteligencia artificial y la lingüística computacional que tiene como objetivo el desarrollo de sistemas capaces de producir textos en cualquier tipo de lenguaje natural (ya sea escrito o hablado) a partir de una representación conceptual generada computacionalmente [8, 9, 10, 11].

2.1. NLG vs NLU

Dentro de la lingüística computacional se encuentra el Procesamiento del Lenguaje Natural cuyos dos grandes bloques o componentes que lo conforman son la Comprensión del Lenguaje Natural (Natural Language Understanding, NLU) y el NLG. En este caso, el NLU se centra en la creación de sistemas con la capacidad de entender cualquier tipo de lenguaje humano. Es fácil ver la estrecha relación entre ambos subcampos. De hecho, muchos autores los presentan uno como el inverso del otro, ya que el NLU se encarga del proceso de mapear el lenguaje natural en representaciones interpretables computacionalmente mientras que el NLG hace el proceso inverso al transformar dichas representaciones internas de la máquina a cualquier tipo de idioma escrito o hablado.

El hecho de trabajar en procesos inversos pero similares hace que NLU y NLG presenten ciertos elementos en común. Dichos elementos aparecen porque tienen una base teórica en común que se apoya en los modelos computaciones del lenguaje. Además, es fácil que ambos subcampos aparezcan de alguna manera como componentes complementarios dentro de un sistema, como es el caso de este trabajo, donde NLU y NLG son dos de los componentes del EMPATHIC-VC.

Sin embargo, a pesar de sus similitudes y de ser procesos prácticamente inversos, no se puede construir un sistema bidireccional que permita trabajar en ambos sentidos correctamente, ya que NLU y NLG presentan retos o problemas de importancia en diferentes cuestiones además de otras diferencias principales. Empezando con las diferencias a nivel de dificultades que cada componente presenta, el NLU puede que tenga que tratar con frases mal escritas o mal formadas como entrada para interpretar. Dichas frases en ningún caso serían generadas por un NLG que trabaje correctamente. Otra diferencia importante es que para el NLU el orden en el que genera las salidas semánticas no es importante, mientras que para el NLG el orden es importante ya que es importante tanto el formato como el contenido de la frase generada. Con todo esto, la más importante de todas las características que dificultan que los dos componentes trabajen como un sistema bidireccional es que las etiquetas que genera el NLU para interpretar el lenguaje (ya que el proceso

Módulos	Fase de contenido	Fase estructural
Macroplanificación o planificación del documento	Selección de contenido	Estructuración del documento
Microplanificación	Lexicalización; Generación de expresiones referenciales	Agregación de sentencias
Realización	Realización lingüística	Realización estructural

Tabla 2.1: Módulos y fases de un NLG

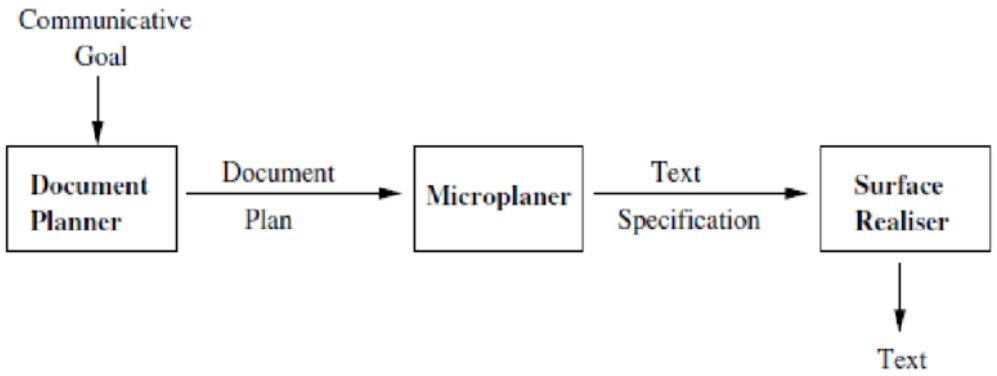

Figura 2.1: Fases de un NLG estándar [13]

de NLU se puede tratar como un problema de etiquetado) no son las entradas que el NLG necesita para poder generar textos adecuados para cualquier aplicación. Este hecho ha sido considerado a la hora de construir el VC de este proyecto y es por eso que las etiquetas elegidas por Montenegro et al. [12] para las representaciones semánticas que utiliza su módulo de NLU difieren de las elegidas para el NLG, como se verá posteriormente.

2.2. Fases de un NLG

A la hora de describir la forma de actuar de un generador de lenguaje natural se puede dividir dicho proceso en distintas fases. Dichas fases fueron presentadas por Ehud Reiter y Robert Dale en el año 2000 [8] y a día de hoy se sigue aceptando que todo NLG presenta dichas fases [11, 9]. Como se podrá comprobar a lo largo de este capítulo, las fases aparecerán de forma más o menos explícita dependiendo del tipo de enfoque que se utilice a la hora de implementar el NLG.

En dicha descomposición del proceso se definen tres módulos diferentes (figura 2.1), que no tienen que estar explícitamente separados dentro del NLG, con un total de siete fases distintas (tabla 2.1). Dichos tres módulos son los de macroplanificación, microplanificación y realización. Todos los módulos presentan fases que se dividen en a nivel de contenido y a nivel estructural.

Comenzando por la *planificación del documento*, el objetivo de esta fase es producir las especificaciones sobre el contenido y la estructura del texto. Para determinar dichas especificaciones se debe atender a nivel de contenido sobre qué información es importante para lo que se quiere comunicar teniendo en cuenta la función comunicativa a expresar, la información sobre el receptor final (qué información ya sabe y cuál es más importante darle a conocer), la información que se tiene dada en la entrada por la fuente o algún tipo de restricción, como podría ser un número máximo de palabras o caracteres. A esta fase es a la que determinamos *selección de contenido*. En cuanto a la *estructuración del documento*, está directamente relacionada con el orden en el que se

va a dar la información, ya que una adecuada generación no consiste siemplemente en agregar un conjunto de informaciones generadas en un orden al azar. Sin embargo, dicha fase va más allá de lo que es la propia secuencialidad en la presentación de la información, ya que tiene como objetivo crear una jerarquía o estructura que establezca las relaciones entre las distintas informaciones a generar.

En algunos casos, las especificaciones dadas por el módulo anterior sirven para definir completamente el contenido y la estructura del texto. Sin embargo, en otros casos es mejor dejar algún tipo de libertad a la hora de determinar ciertas decisiones. Dichas decisiones están basadas en el conocimiento del lenguaje a utilizar. Así, el módulo de *microplanificación* tiene las tareas de seleccionar las palabras y referencias a utilizar así como decidir cómo agrupar las distintas informaciones en frases y párrafos. En este módulo, se distinguen tres fases, dos a nivel de contenido y una a nivel estructural. En la primera fase referente al contenido, la de *lexicalización*, se eligen las palabras que se van a utilizar para expresar el contenido seleccionado en el módulo anterior. Por su parte, la función de la *generación de expresiones referenciales* está ligada a como se va a hacer referencia a las distintas entidades que aparezcan en el texto. Teniendo en cuenta que no es lo mismo hacer referencia a una entidad por primera vez que las siguientes veces, el reto detrás de esta fase es que la referencia a cada entidad sea lo más clara posible para poder distinguirla de cualquiera de las otras entidades. Finalmente, a nivel estructural en la microplanificación se debe decidir cómo agregar los distintos mensajes procedentes del plan del documento para que el resultado sea un conjunto coherente de mensajes. De este modo, lo que se realiza en la fase de *agregación* es realizar un mapeado de informaciones decidiendo qué información va en cada frase y a su vez qué conjunto de frases forman parte de cada párrafo.

El último módulo es el de *realización*. Este módulo transforma todas las posibles especificaciones abstractas que se hayan decidido en las fases anteriores en los textos reales finalmente generados. En la *realización lingüística* se utilizan reglas gramaticales, morfosintácticas y ortográficas para determinar el texto elegido sin tener en cuenta ya ningún aspecto externo relacionado con el dominio donde se vaya a utilizar el texto. A nivel estructural, lo que se realiza en la fase de *realización estructural* es hacer que a la hora de representar el texto lo que se haya decidido agrupar en un párrafo aparezca como tal y lo que aparezca en párrafos separados lo haga también.

En lo que se refiere a este trabajo, el hecho de que su aplicación sea dentro de un SDS hace que dichas fases se simplifiquen ligeramente, ya que a nivel estructural nunca vamos a dividir el texto en párrafos sino que será una sucesión de frases presentadas de modo coherente. En estos casos, partiendo de esta base teórica se suele decir que un NLG estándar dentro de un SDS se divide en dos fases principales: planificación de sentencias, centrado en la representación abstracta de dicha sentencia o conjunto de sentencias, y la realización, con el mismo objetivo que antes, que no es otro que el de transformar las representaciones abstractas en frases [14, 15].

2.3. Tipos de entradas de un NLG

Una primera clasificación que se puede hacer para establecer los diferentes tipos de NLG es basándose en el formato de las representaciones de las que se partan. Aunque las entradas de un NLG pueden ser muy variadas, ya que pueden pueden ir desde grafos [16] a parejas slot-values [17] e incluso el uso de datos de un tipo totalmente alejado del texto como podrían ser imágenes [15], la clasificación a este nivel se divide en dos tipos: de dato a texto (D2T) y de texto a texto (T2T) [11].

Los generadores D2T, también conocidos como generadores de concepto a texto, son los más

típicos dentro del NLG. De hecho, los tres ejemplos presentados anteriormente se encuadrarían dentro de este tipo. Los datos normalmente suelen ser numéricos pero también puede ser cualquier otro tipo de dato estructurado sacado de corpus etiquetados, bases de datos u otras fuentes. La explicación a que este tipo de sistemas sean más frecuentes no es otra que la representación interna que puede generar un ordenador solo va a ser textual en el caso de que dicho texto se haya proporcionado al ordenador desde otra fuente.

Por su parte, los generadores T2T son aquellos que tienen como entrada textos u oraciones aisladas. La aplicabilidad de estos generadores suele alejarse en algunos casos de la idea del NLG ya que más que crear texto nuevo se aprovechan del texto de entrada para realizar resúmenes o corrección de los textos, aunque también los procesos de traducción automática se engloban aquí, los cuales tienen una alta parte generativa. Sin embargo, es posible encontrar sistemas T2T dentro del mismo dominio que un D2T como en el caso de las recomendaciones sobre restaurantes donde podemos encontrar un ejemplo de D2T en el generador MATCH que dentro de un SDS genera recomendaciones a partir de un conjunto de datos multiatributo [18], mientras que Have2eat hace recomendaciones de restaurantes a base de resumir las opiniones del usuario sobre el restaurante [18], por lo que quedaría englobado dentro de la clasificación de T2T.

Como se verá posteriormente, el NLG a diseñar para EMPATHIC se clasificaría como un D2T. El hecho de que esté integrado en un SDS hace que reciba del gestor de diálogo representaciones conceptuales de lo que debe responder. Dichas representaciones en ningún caso son texto.

2.4. Aplicaciones del NLG

Al igual que se pueden clasificar los NLG en función de su entrada también puede hacerse a partir de una perspectiva de su aplicabilidad. Solo atendiendo a la última edición de la International Conference of Natural Language Generation (INLG 2018), la conferencia más importante a nivel de NLG, se puede ver que las aplicaciones del NLG son muy distintas: descriptor de productos para una plataforma de comercio electrónico [19], generadores de preguntas[20], sistemas con la tarea de hacer referencias a objetos de manera precisa [21], generadores de texto hablado para SDS [22], etc. En este caso es difícil encontrar un criterio común a la hora de establecer una clasificación en base a su aplicación debido a la alta variabilidad que hay tanto en tareas como en dominios [11, 23, 24, 8]. De este modo. se ha decidido seleccionar las aplicaciones más comunes:

- **Traducción automática:** Esta tarea tiene como función traducir textos de un idioma a otro. Muchos de los conceptos utilizados en esta tarea son extrapolables a la tarea de NLG, ya que la generación de lenguaje natural se puede entender como un proceso de traducción desde el formato de entrada hasta el idioma en lenguaje natural escogido.

- **Generación de resúmenes:** Estos generadores realizan la función de expresar de forma más abreviada la información textual obtenida de una o varias fuentes.

- **Generación de textos simplificados:** Son sistemas diseñados con el objetivo de escribir textos de una forma más clara o sencilla que la original. Normalmente este tipo de sistemas se implementan atendiendo a la razón de que el receptor va a ser una persona con una capacidad de compresión lectora menor.

- **Generación de textos informativos:** La finalidad asociada a estos generadores es dar una representación en lenguaje natural de datos contenidos en el ordenador: datos objetivos. De este modo, lo que se busca es simplificar la interpretación de dichos datos.

- **Generación de recomendaciones:** Dentro del NLG, una de las funciones cada vez más típicas es el uso de estos sistemas para realizar recomendaciones, por ejemplo apoyadas en las opiniones dadas por otros usuarios. Como ya hemos mencionado antes dicho recomendador puede utilizar ideas similares al de la generación de resúmenes si cuenta con opiniones en formato texto de los usuarios o utilizar otras técnicas de NLG si el punto de partida son valoraciones numéricas.

- **Generación de sistemas de diálogo:** Son sistemas de generación de texto que aparecen en el contexto de interacción entre máquina y persona. Dicho contexto hace que al sistema al interactuar con un usuario a la hora de generar las respuestas tenga que tener en cuenta el estado de la conversación.

- **Generación de textos persuasivos:** Se le da dicha consideración a todo texto que trate de modificar de alguna forma al usuario o su estado de ánimo.

Bajo estas consideraciones, el sistema de este trabajo entraría dentro de los dos últimos tipos de sistema. Como ya se mencionó en el capítulo 1, el NLG forma parte de un SDS, lo que conlleva que es un sistema de diálogo, con el añadido de que utiliza lenguaje natural hablado. Por otro lado, viendo que dentro de los objetivos de EMPATHIC está incentivar un estilo de vida que derive en un envejecimiento saludable, el generador desarrollado no puede dejar de ser un generador persuasivo.

2.5. NLG en Sistemas de diálogo Hablado

El hecho de que el EMPATHIC-VC sea un SDS implica que es un sistema diseñado para una interacción máquina-humano capaz de mantener una conversación por voz [10, 1]. Estos sistemas cada vez son más utilizados y prueba de ello es que grandes empresas presentan tecnologías de este tipo: Siri de Apple[*] o Cortana de Microsoft[**]. Pero no solo se encuentran dentro de las grandes empresas, sino que dentro del mundo académico también se esta investigando sobre ello: AdApt [25], un sistema multimodal (toma como entrada la imagen de un mapa y la voz del usuario) desarrollado para proporcionar información sobre los apartamentos a la venta en un barrio sueco, o Let'sGo [26], centrado en mejorar la calidad de un sistema telefónico automático de información sobre buses para posibles receptores como gente no nativa o personas mayores, son ejemplos de ello.

Todos los SDS presentan una arquitectura similar compuesta por diferentes módulos que permiten ir desde el reconocimiento de lo que el usuario ha dicho hasta la generación de una respuesta adecuada mediante voz. Dichos componentes habituales (figura 2.2), también presentes en la arquitectura EMPATHIC (figura 1.1), son los que siguen:

- **Automatic Speech Recognition (ASR):** Generar una transcripción a partir del audio capturado de lo que ha dicho el usuario. Dicha transcripción puede venir acompañada de su grado de confianza, determinado por la confianza en la calidad del texto generado, así como de su tiempo de inicio y la duración.

- **Natural Language Understanding (NLU):** El NLU a partir del texto generado por el ASR trata de capturar los conceptos que el usuario expresa en cada turno. Dichos conceptos se extraen mediante la asignación de unas etiquetas semánticas a la transcripción dada. En el caso de

[*] https://www.apple.com/es/siri/

[**] https://www.microsoft.com/es-es/windows/cortana

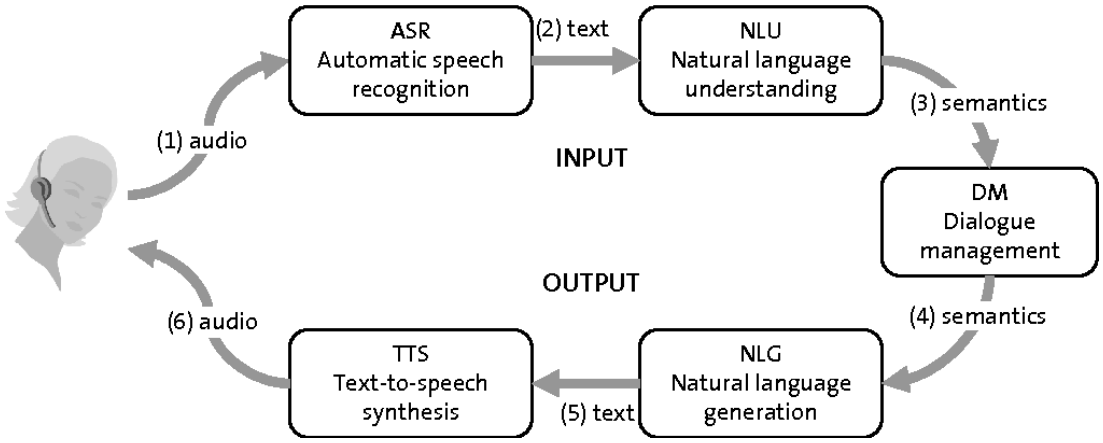

Figura 2.2: Arquitectura típica de los SDS [27]

EMPATHIC, dicho módulo asigna una etiqueta de la intención del usuario, otra del tema tratado y además se hace un proceso de Entity Recognition. Este último elemento es importante para el NLG ya que las entidades reconocidas por el NLU son utilizadas como valores de los atributos que el Dialog Manager envía al NLG, como ya se había explicado anteriormente.

- **Dialog Manager (DM):** Se trata del gestor de diálogo y está encargado de decidir la acción a realizar en cada paso teniendo en cuenta el estado de la conversación, el cual se actualiza con cada llegada de información y que tiene en cuenta todo el historial de la conversación así como los datos del usuario. En los SDS habituales la información actualizadora del estado se extrae de las etiquetas semánticas generadas por el NLU, pero en algunos casos puede haber otras informaciones que pueden afectar a la toma de decisión del DM. En EMPATHIC, otra de las fuentes de información es el módulo de emociones. La decisión tomada se representa mediante un DA con el cual se le informa al NLG sobre la respuesta que tiene que generar.

- **Natural Language Generator (NLG):** Su función dentro de los SDS es generar el texto de respuesta a través de la información conceptual que recibe por parte del DM en forma de DA. En algunos casos, envía más información al siguiente módulo (la que previamente el DM le ha enviado) además del propio texto pero en esa información el NLG no genera ningún tipo de transformación, sólo actúa de puente. A estas alturas, en EMPATHIC la información relevante que recibe el módulo de síntesis de voz por parte del NLG aparte del texto y el idioma es el género del agente y la polaridad que determinan el tipo de voz utilizada en cada caso.

- **Text to Speech (TTS):** El último componente de la cadena habitual de los SDS tiene la tarea de generar el audio de respuesta a partir del texto y la información adicional generados en el NLG.

La importancia del NLG dentro de este tipo de sistemas es vital, sobretodo en cuanto a la aceptación por parte del usuario, ya que un sistema que genere respuestas lingüísticamente incorrectas, poco naturales o con una variabilidad de respuesta muy baja hace que la valoración de los usuarios baje. Como ya comentan Wen et al. la calidad del generador dentro de los sistemas de diálogo es vital en términos de la usabilidad y la aceptación que tenga [28].

El hecho de implementar un generador para un SDS presenta una serie de diferencias con la forma de trabajar con otro tipos de NLGs [29]. Esto establece una serie de retos que se presentan a continuación:

- La primera diferencia es clara: la forma de expresarse en formato escrito no es la misma que de forma hablada. Las sentencias utilizadas en una conversación suelen ser más cortas y menos restrictivas a nivel gramatical que lo que son a nivel escrito. De este modo, el uso de expresiones simples y cortas debe ser una de las premisas para dar lugar a una comunicación más efectiva en este contexto.

- Normalmente dichos sistemas de diálogo se presentan dentro de un dominio específico, lo que reduce el léxico o vocabulario a utilizar. De este modo, se deben desarrollar sistemas con un léxico apropiado para el dominio y tarea requeridos.

- El NLG en estos sistemas se presenta como parte de un conjunto de módulos de los cuales todos tienen su importancia a la hora de un buen funcionamiento y una buena aceptabilidad por parte del usuario. Es por ello que el NLG debe estar adaptado al resto de módulos tratando de maximizar las virtudes del resto de módulos y tratando de ocultar en la medida de lo posibles sus limitaciones.

2.6. Enfoques para el NLG

Hasta ahora hemos hecho hincapié tanto en la entrada como en la salida (sus aplicaciones) a la hora de clasificar un NLG pero la más importante de las clasificaciones es aquella que deriva del tipo de modelo implementado. De este modo, aunque no es la única división posible para el tipo de modelos [23] se va a considerar que los modelos se agrupan dentro de dos tipos distintos: basados en reglas o en el conocimiento y full-data-driven o basados en datos a pesar de que existe la posibilidad de combinar ambos tipos en lo que se conoce como tipo híbrido o semi-data driven. [11, 10].

2.6.1. Basados en reglas: plantillas vs gramáticas

Los modelos basados en reglas o conocimiento se caracterizan por el hecho de que sus técnicas se nutren de elementos lingüísticos como bases de conocimiento léxico, reglas, plantillas que deben construirse manualmente. Debido a dichas características, sus principales ventajas son su robustez, precisión y la escasa necesidad de datos. Sin embargo, se suelen presentar como modelos limitados por ser extremadamente rígidos, repetitivos y con poca capacidad de ser reutilizados en distintos dominios [30]. Es posible encontrar diferentes enfoques incluso dentro de estos modelos de NLG.

El enfoque más rígido y más simple es aquel cuyo proceso consiste básicamente en asignar a cada entrada una salida o conjunto de salidas completamente definidas, lo que se conoce como *canned text*. Se trata de una asignación directa sin representaciones intermedias. De este modo, ante una entrada lo único que se haría es seleccionar una de las salidas en caso de que haya más de una. Aunque no está muy ampliamente utilizado hay ejemplos de SDS como NPCEditor [31] que trabaja de este modo al tener asignadas las respuestas a cada tipo de pregunta posible.

Los NLG basados en plantillas (figura 2.3) aplican una idea similar, ya que básicamente la única diferencias es que a cada entrada no le asignamos una frase totalmente formada sino que se le asigna un plantilla o conjunto de plantillas. las plantillas son estructuras lingüísticas que pueden contener huecos. Dichos huecos hay que rellenarlos para obtener una frase de salida totalmente formada. Los huecos habitualmente contienen atributos que se reemplazan por el valor que se les viene asignado en la entrada. Estos sistemas siguen muy presentes en la mente de los diseñadores y buena prueba de ello es la cantidad de generadores con dicho enfoque que se han creado en los

Figura 2.3: Generación basada en plantillas

últimos años [32, 33, 34].

Los NLG basados en la gramática son el último tipo de sistemas basados en reglas. En este caso, a diferencia de lo anteriormente explicado, la generación de texto tiene una implicación más creativa ya que en este caso el proceso está basado en establecer unas reglas gramáticas a las que se les añade un vocabulario para formar las frases. Este tipo de sistemas es el que más se adapta a lo que definiríamos como NLG estándar ya que sigue la estructura modular definida en la sección 2.2 de manera más fidedigna que el resto de enfoques. A pesar de ser un enfoque bastante tradicional, no deja de ser un enfoque bastante utilizado [35, 36, 9].

Algunos autores como Deemter et al. [37] argumentan que las diferencias entre estos dos últimos enfoques no representan una diferencia real ya que se puede establecer una equivalencia entre los procesos en ambos casos. Por un lado, el proceso de selección de plantillas es una primera fase de la generación que viene a simular las dos primeros módulos de la estructura estándar. Mientras que la fase de realización vendría a ser el reemplazo de los huecos de las plantillas por los respectivos valores de los atributos. Sin embargo, la sensación personal es que esa diferencia sí que es real a pesar de esas posibles equivalencias. De hecho, una explicación propia de lo que representa cada enfoque es que un sistema basado en la gramática lo que se está haciendo es enseñar al sistema un idioma ya que se le están dando las reglas y dando un vocabulario para el propio sistema aprenda a formar frases en el lenguaje correspondiente, mientras que los otros dos sistemas aprenden a decir una serie de frases concretas que saben generar pero sin conocer el idioma que utilizan.

En contraposición a lo expuesto por Deemter, en la bibliografía se citan diferencias reseñables entre estos tres enfoques. Se centran en temas de mantenimiento del sistema, escalabilidad, calidad de la salida y multilingüismo [37, 32, 38]. La idea principal es que los sistemas basados en plantillas presentan desventajas en todas ellas. Sin embargo, ciertos razonamientos son discutibles. A nivel de mantenimiento y escalabilidad, la idea por la cual se entiende que las plantillas son inferiores es que, una vez el sistema basado en gramática tiene implementada las reglas básicas de un idioma, el adaptarlo a un nuevo dominio requiere menos cambios que el hecho de tener que generar nuevas plantillas para el mismo sistema o para un nuevo sistema o dominio. Pero en realidad, el hecho de enseñar una gramática tan completa a un sistema no es nada sencillo y la realidad es que finalmente dichos sistemas suelen quedar muy restringidos al dominio en el que trabajan, así que finalmente a nivel de mantenimiento sí que dicha ventaja puede ser más considerable pero a nivel de escalabilidad dicha ventaja no es tan real ya que un nuevo dominio suele suponer grandes cambios también en los sistemas basados en gramáticas. En cuanto a la calidad de la salida, se menciona en ciertos artículos que el hecho de trabajar en distintas fases permite al NLG estándar ir mejorando gradualmente la salida del sistema pero realmente con las plantillas se asegura que la salida vaya a ser correcta con mayor seguridad, ya que la posible creatividad del sistema queda reducida. Además, toda aquella calidad en cuanto a tema de agregación de sentencias o ciertos aspectos en la formación de las frases también se puede conseguir con las plantillas a través de un buen uso de los atributos. En cuanto al multilingüismo, se ha visto que para uno de

los sistemas grammar-based más utilizados como es SimpleNLG, para crear un sistema bilingüe inglés-francés se han necesitado 5 meses solamente para adaptar el sistema ya construido en inglés al francés, mientras que una traducción de las plantillas utilizadas en un sistema a otro idioma requeriría una cantidad de tiempo mucho menor.

Con todo esto, los NLG basados en reglas pueden ser ideales para ciertas tareas y dominios e incluso ante la falta de datos pueden ser buenas alternativas para un SDS debido a su robustez y la capacidad de construir respuestas lingüísticamente correctas. Sin embargo, la tendencia actual en los NLG dentro de los SDS va encaminada a uso de sistemas basados en datos debido a la aparición de ciertas técnicas y al aumento de la cantidad de datos.

2.6.2. Sistemas basados en datos

Los sistemas basados en datos, también conocidos como estadísticos, son una alternativa que cada vez está tomando más fuerza dentro de la generación de lenguaje de los SDS. La principal característica de estos sistemas es que generan sus modelos a partir de un corpus de entrenamiento, lo que permite la generación sin la codificación a mano de las reglas del idioma a generar.

El NLG es una de las últimas áreas en la que se introdujeron los métodos estadísticos. De hecho, la aparición de dichos modelos data de 1998, cuando Langkilde y Knight utilizaron los modelos de lenguaje para realizar transformaciones a nivel de palabras tras haber aplicado una generación basada en reglas [39, 10, 40]. Sin embargo, el abanico de modelos y técnicas que han sido propuestas ya es muy amplio y en muchos casos mejoran el funcionamiento de los sistemas basados en reglas. Es por ello que estos modelos han sido probados como una alternativa viable y que ha llamado la atención dentro de la investigación, donde se distinguen dos líneas de trabajo. La primera introduce la estadística como reevaluación de las frases candidatas establecidas por otro generador, con modelos de lenguaje entrenados u otros tipos de modelos basados por ejemplo en la valoración de los usuarios de la calidad de las frases. Por su parte, la segunda línea de investigación presente a nivel estadístico consiste en el entrenamiento de modelos que tratan de encontrar unos parámetros adecuados para maximizar una función objetivo, la cual está relacionada con la calidad de las salidas.

Aunque actualmente la cantidad de datos para la generación natural no se puede considerar aún muy amplia sí que ha habido un aumento notable en la cantidad de datos, lo que posibilita el uso de dichas técnicas [41]. Entre las técnicas más utilizadas destacan los modelos del lenguaje, que son un mecanismo para definir la estructura del lenguaje basado en restringir adecuadamente las secuencias de palabras más probables. Se le asigna alta probabilidad a sentencias correctas y muy baja a sentencias incorrectas. A parte de usarlos como generador, una idea muy extendida es utilizarlos como elemento corrector de la frase generada con otro tipo de generador. Sin embargo, son demasiado generales, lo que limita su uso como generador dentro de un SDS.

Una de las tendencias actuales a la hora de implementar NLG dentro de un SDS es el uso de técnicas basadas en deep learning [22, 42]. Se entiende que con ellas se puede mejorar la percepción del usuario sobre el sistema gracias a su capacidad de generalizar mejor. Además, a nivel de desarrollo necesita de menos esfuerzo manual a la hora de ampliar a nuevos dominios o incrementar el dominio actual. Estos sistemas son denominados end-to-end (E2E) por su capacidad de aprender de forma conjunta las distintas fases de un NLG estándar a partir de los datos y, a su vez, generar en un solo paso el texto generado a partir de la representación conceptual sin representaciones intermedias. Dichos E2E hacen uso de redes neuronales permitiendo generar texto a partir

de diferentes tipos de datos estructurados.

Uno de los sistemas de redes neuronales más utilizado son las redes neuronales recurrentes (Recurrent Neural Networks, RNNs) con celdas LSTMs o GRUs os, los cuales integrados en un entorno de seq2seq, han dado buenos resultados en distintos procesos como traducción automática, modelización de la conversación y, por supuesto, NLG . Tal es el avance en este campo, que las redes neuronales basadas en seq2seq que trabajan a nivel de palabra tienen su equivalente en lo que se conoce como las char2char [30].

2.7. Etiquetado de actos de diálogo

El etiquetado de los actos de diálogo (Dialog Acts, DAs) es una tarea que está estrechamente unida al módulo de NLG, el cual recibe estas representaciones conceptuales de las respuestas por parte del gestor de diálogo para generar los turnos del VC. Así, la selección de dichas representaciones está relacionada directamente con las necesidades del NLG. Un DA es una etiqueta que recoge tanto la función comunicativa como el contenido semántico de un elemento de un diálogo o discurso. El contenido semántico sería la información que se suministra, mientras que la función comunicativa correspondería al modo en el que se presenta dicha información (afirmación, pregunta, expresando una duda,...) [43].

Basándose en la teoría de los actos del habla de Austin [44], muchos estudios se centran en los actos de diálogo para analizar el fenómeno del diálogo, para establecer sistemas de anotación óptimos de los mismos o para su uso en el diseño de sistemas de diálogo [43, 45, 46, 12]. La idea inicial de establecer una etiqueta o un conjunto de etiquetas para codificar una frase o turno del diálogo ha derivado en una teoría con un enfoque en el cual a partir de los datos se trata conseguir un modelado computacional del uso del lenguaje interactivo. En todos estos análisis, lo primero que hay que determinar es cuál es la segmentación óptima del diálogo. En algunos casos se usa el turno como elemento básico del diálogo. Sin embargo, algunos turnos pueden llegar a ser muy largos y , lo que es más importante, tener más de una función comunicativa. Es por ello, que en muchos casos se elige como elemento del diálogo lo que se conoce como sentencias, la unidad mínima de información dentro del diálogo. La relación entre turno y sentencia en algunos puede ser uno a uno pero no tiene por qué serlo necesariamente.

2.7.1. Sistemas de anotación de DAs

Antes de definir el sistema de anotación que se ha utilizado en esta tarea, se ha hecho un análisis previo de los sistemas existentes. A lo largo de los años se han diseñado múltiples conjuntos de DA con diferentes taxonomías. Los diferentes conjuntos vienen caracterizados por el tipo de comunicación anotada, el dominio de la tarea y la dimensionalidad de la anotación, entre otras cosas.

El tipo de comunicación hace referencia a si los participantes de la comunicación interactúan en un tiempo común (comunicación síncrona) o lo hacen de una forma en la que no coinciden en el tiempo (comunicación asíncrona). De este modo, para comunicaciones síncronas tendríamos una conversación *in situ*, mediante vídeollamada o mediante un servicio de mensajería instantánea. Por su parte, libros, correos electrónicos, blog u otros formatos de comunicación escrita entrarían dentro de lo asíncrono. Generalmente este último tipo de comunicación presenta una estructura más fácil de análisis al no presentar dudas, rectificaciones, conversaciones solapadas o

ruido.

Con respecto a este último tipo de comunicación, se ha analizado el recurso Pen Discourse Tree Bank (PDTB), que cuenta con un conjunto de anotaciones realizadas sobre el Wall Street Journal Corpus [47]. Dichas anotaciones marcan la relación entre sentencias dentro de un discurso, es decir, tratan de determinar la forma en la que se conectan unas frases con otras, ya sea con conectores, elementos que se repiten en ambas frases u otro tipo de conexión. Sin embargo, este tipo de etiquetado no se adapta ni al tipo de datos con el que se cuenta ni el propio concepto de cómo se realiza la anotación es útil para este trabajo. Es por ello que se ha pasado a analizar taxonomías que trabajen sobre comunicación síncrona.

Las dos taxonomías de actos de diálogo síncronas analizadas sí que han servido de ayuda a la hora de establecer el conjunto de etiquetas utilizado en este trabajo. En ambos casos, se trata de taxonomías de dominio abierto ya que se han diseñado con la idea de proporcionar un sistema de anotación para múltiples propósitos. La primera metodología de anotación es una adaptación de la metodología DAMSL al Switchboard (SWBD) corpus que se conoce como SWBD-DAMSL [48, 45] y la segunda, DIT++, que surge como un esfuerzo por parte de la organización ISO de establecer un estándar de anotación [46, 43].

SWBD-DAMSL

SWBD-DAMSL es un conjunto de etiquetas diseñado por la Universidad de Stanford [45], que tiene como base la metodología de anotación Discourse Annotation and Markup System of Labeling (DAMSL)[48]. DAMSL fue creado por James Allen y Marc Core y tenía como idea definir un esquema de anotación que pudiera ser utilizado en un amplio rango de diálogos. Dicho esquema trabaja sobre tres capas principales, *Forward Communication Function*, *Backward Communication Function* y *Feature sentencia*. La primera es una capa que contiene las etiquetas típicas de la teoría del acto del habla (sentencia informativa, opinión, pregunta, orden,...). La segunda hace referencia a todas aquellas funciones comunicativas relacionadas con el diálogo previo, como aceptar una proposición, dar una respuesta a una pregunta o confirmar que se entiende algo. Por su parte, la última capa se centra en analizar el contenido y el formato de la sentencia como puede ser indicar la temática de la sentencia, tipos de formatos (correcto, inacabado o no interpretable) y tipo de sentencia (convencional o exclamativa).

SWBD-DAMSL presenta diferencias con este formato que justifica a partir de la adaptación del sistema al corpus Switchboard. Dicho corpus no presenta grandes restricciones ya que contiene de un conjunto de 1555 conversaciones telefónicas humano-humano elegidas de forma aleatoria y donde el tema no está predefinida de ningún modo. En cuanto a la segmentación de los diálogos, tanto el sistema primigenio como el posteriormente desarrollado hacen uso de las mencionadas sentencias. Sin embargo, a pesar de aprovechar el conjunto de etiquetas definidas en DAMSL, el sistema pasa de ser un sistema multidimensional de etiquetas con posibilidad para asignar una etiqueta por capa a un sistema con 42 etiquetas mutuamente exclusivas (figura 2.4). En este proceso de selección de etiquetas, inicialmente se han escogido las etiquetas atendiendo a dos factores: lingüísticamente interesantes y que sean detectables de forma fiable. De dicha selección se obtenía un sistema de 50 etiquetas básicas cuya combinación daba un total de más de 220 etiquetas. Finalmente, para poder construir un sistema que permitiera una mayor posibilidad de acuerdo entre los anotadores así como suficientes datos por etiqueta para construir modelos con ellas se hizo la reducción a los 42 actos de diálogo finales.

Tag	Example	%
STATEMENT	*Me, I'm in the legal department.*	36%
BACKCHANNEL / ACKNOWLEDGE	*Uh-huh.*	19%
OPINION	*I think it's great*	13%
ABANDONED / UNINTERPRETABLE	*So, -/*	6%
AGREEMENT / ACCEPT	*That's exactly it.*	5%
APPRECIATION	*I can imagine.*	2%
YES-NO-QUESTION	*Do you have to have any special training?*	2%
NON-VERBAL	*<Laughter>,<Throat_clearing>*	2%
YES ANSWERS	*Yes.*	1%
CONVENTIONAL-CLOSING	*Well, it's been nice talking to you.*	1%
WH-QUESTION	*What did you wear to work today?*	1%
NO ANSWERS	*No.*	1%
RESPONSE ACKNOWLEDGMENT	*Oh, okay.*	1%
HEDGE	*I don't know if I'm making any sense or not.*	1%
DECLARATIVE YES-NO-QUESTION	*So you can afford to get a house?*	1%
OTHER	*Well give me a break, you know.*	1%
BACKCHANNEL-QUESTION	*Is that right?*	1%
QUOTATION	*You can't be pregnant and have cats*	.5%
SUMMARIZE/ REFORMULATE	*Oh, you mean you switched schools for the kids.*	.5%
AFFIRMATIVE NON-YES ANSWERS	*It is.*	.4%
ACTION-DIRECTIVE	*Why don't you go first*	.4%
COLLABORATIVE COMPLETION	*Who aren't contributing.*	.4%
REPEAT-PHRASE	*Oh, fajitas*	.3%
OPEN-QUESTION	*How about you?*	.3%
RHETORICAL-QUESTIONS	*Who would steal a newspaper?*	.2%
HOLD BEFORE ANSWER/ AGREEMENT	*I'm drawing a blank.*	.3%
REJECT	*Well, no*	.2%
NEGATIVE NON-NO ANSWERS	*Uh, not a whole lot.*	.1%
SIGNAL-NON-UNDERSTANDING	*Excuse me?*	.1%
OTHER ANSWERS	*I don't know*	.1%
CONVENTIONAL-OPENING	*How are you?*	.1%
OR-CLAUSE	*or is it more of a company?*	.1%
DISPREFERRED ANSWERS	*Well, not so much that.*	.1%
3RD-PARTY-TALK	*My goodness, Diane, get down from there.*	.1%
OFFERS, OPTIONS & COMMITS	*I'll have to check that out*	.1%
SELF-TALK	*What's the word I'm looking for*	.1%
DOWNPLAYER	*That's all right.*	.1%
MAYBE / ACCEPT-PART	*Something like that*	<.1%
TAG-QUESTION	*Right?*	<.1%
DECLARATIVE WH-QUESTION	*You are what kind of buff?*	<.1%
APOLOGY	*I'm sorry.*	<.1%
THANKING	*Hey thanks a lot*	<.1%

Figura 2.4: Etiquetas de SWBD-DAMSL [45]

Dimension	*Dimension-specific ommunicative functions*	*Typical expressions*
Task/Activity	OpenMeeting, CloseMeeting; Appoint, Hire, Fire	domain-specific fixed expressions
Auto-Feedback	PerceptionNegative	*Huh?*
	EvaluationPositive	*True.*
	OverallPositive	*OK.*
Allo-Feedback	InterpretationNegative	*THIS Thursday.*
	EvaluationElicitation	*OK?*
Turn Management	TurnKeeping	final intonational rise
	TurnGrabbing	hold gesture with hand
	TurhGiving	*Yes.*
Time Management	Stalling	slowing down speech; fillers
Contact Management	ContactChecking	*Hello?*
Own Communication Management	SelfCorrection	*I mean...*
Partner Communication Management	PartnerCompletion	completion of partner utterance
Discourse Structure Management	DialogueActAnnouncement	*Question.*
	TopicShiftAnnouncement	*Something else.*
Social Obligations Management	Apology	*I'm sorry.*
	Greeting	*Hello!, Good morning.*
	Thanking	*Thanks.*

Figura 2.5: Funciones comunicativas de dimensión especifica de DIT++ [43]

DIT++

En 1999, Bunt formuló la Dynamic Theory of Dialogue (DIT) como base para la construcción de una taxonomía multidimensional con aplicabilidad para cualquier dominio y tarea [49]. En dicho artículo, establece una división conceptual entre *Action-Discussion*, que engloba un conjunto de etiquetas en la que los interlocutores negocian ciertas acciones, e *Information-Transfer*, que son aquellas interacciones en las que se realiza algún tipo de intercambio o solicitud de información. En la figura 2.5 se puede ver qué tipo de etiquetas quedan englobadas en cada grupo.

Con los conceptos de esta versión de trasfondo, el propio Bunt desarrolló DIT++ con el objetivo de desarrollar un estándar en la anotación de los actos de diálogo [46]. Es por ello que con DIT++ se trató de diseñar un modelo de etiquetado capaz de capturar tanto la función comunicativa como el contenido semántico. Con esta finalidad, la taxonomía presenta dos elementos: las funciones comunicativas y las dimensiones. Cada dimensión agrupa un conjunto de funciones comunicativas (figura 2.5) y debido a la multidimensionalidad a cada DA se le pueden asignar múltiples etiquetas aunque solo una etiqueta por dimensión.

Se pueden distingui dos tipos de funciones comunicativas: las funciones de propósito general (figura 2.6), que pueden ser utilizadas en cualquier dimensión, y las funciones de dimensión especifica (figura 2.5), que son exclusivas de una única dimensión. En el caso de las de propósito general queda claramente reflejado que la teoría DIT está presente, ya que las funciones comunicativas se dividen en Information Transfer Functions (figura 2.6a) y Action Discussion Functions (figura 2.6b, mientras que las funciones comunicativas de dimensión especifica quedan encuadradas dentro de las 10 dimensiones que presenta DIT++. A la hora de realizar una anotación se hace a través de la función o funciones comunicativas correspondientes. Si la función es de dimensión especifica ya es suficientemente descriptiva, pero en caso de asignar una de propósito general se debe especificar dentro de qué dimensión.

– *Information Transfer Functions*
  – *Information-Seeking Functions*
    – *Direct Questions*
      – propositional question, set question, alternatives question, check question, etc.
    – *Indirect Questions*
      – indirect propositional question, set question, alternatives question, check question, etc.
  – *Information-Providing Functions:*
    – *Informing Functions:*
      – inform, agreement, disagreement, correction;
      – *Informs with Rhetorical or Attitudinal Functions, such as* elaboration, justification, exemplification.. and warning, threat,..
    – *Answer Functions:*
      – propositional answer, set answer, confirmation, disconfirmation

(a) Information Transfer Functions

– *Action Discussion Functions*
  – *Commissives*
    – offer, promise, address request
    – *other commissives, expressable by means of performative verbs*
  – *Directives:*
    – instruction, address request, indirect request, (direct) request, suggestion
    – *other directives, such as advice, proposal, permission, encouragement, urge,..., expressable by means of performative verbs*

(b) Action Discussion Functions

Figura 2.6: Funciones comunicativas de propósito general [43]

2.8. Evaluación del NLG

La evaluación del NLG presenta ciertas dificultades [23, 50, 10, 11]. Por un lado, las entradas a partir de las cuales se generan textos dentro de un mismo tipo de aplicación pueden ser bastante diferentes lo que dificulta la comparativa entre modelos. Por otro lado, a nivel individual también es difícil establecer un sistema de evaluación óptima, ya que como sucede con otros procesos de NLP en los que la salida es textual, como la generación de resúmenes o la traducción automática, dicha salida tiene múltiples valores óptimos lo que le da una naturaleza subjetiva a la evaluación y le quita certeza al hecho de realizar una comparativa con una salida esperada. Así, ante la falta de una métrica estándar de evaluación, lo habitual es usar varías métricas distintas y analizar su relación.

Dentro de las formas de evaluar la generación, se distinguen dos tipos de métodos en función de la metodología utilizada. Los métodos intrínsecos de evaluación son aquellos en los que se mide la calidad del sistema en función del texto generado, sin ir más allá en los aspectos a tener en cuenta. Mientras que en el caso de los métodos extrínsecos se evalúa el NLG como parte de un todo que es el sistema en el que se integre o aplicación en la que se presente. En un sistema como en el caso de EMPATHIC, la aceptabilidad de los textos generados por parte de los usuarios o la adaptabilidad a los DA generados por el DM podrían ser medidas de este tipo. De hecho, en sistemas reales la opinión de usuarios reales es un factor determinante en la definición final del sistema.

2.8.1. Evaluación automática

A pesar de las dificultades antes mencionadas, existen métodos de evaluación automática que tienen su origen en otros procesos de NLP. Son métricas que han tenido éxito en esos campos y que por su semejanza con el NLG se ha creído que pueden ser adaptables. La mayoría de ellos utilizan una comparativa basada en la coincidencia de n-gramas entre la salida generada y la o las frases referencia que se consideren adecuadas para la representación conceptual dada. Los detalles de algunas de las más utilizadas son los siguientes[51, 22]:

- **BLEU:** Se trata de una métrica que en origen se creó con idea de utilizarse en el campo de la

traducción automática [52]. Como se ha mencionado antes, el cálculo de la métrica tiene su base en el solapamiento de n-gramas:

$$P_n = \frac{ngramas\ comunes}{ngramas\ frase\ generada}$$

donde n=1 es para palabras, n=2 para bigramas y así sucesivamente. El cálculo de BLEU utiliza una media de P_n a la que a cada tipo de solapamiento en función del tamaño del n-grama se le asocia un peso, ω_n, que normalmente se suele tomar el mismo para todos ($\omega_n = \frac{1}{N}$):

$$BLEU = BP \sum_{n=1}^{N} \omega_n \log P_n$$

Lo habitual es tomar N=4 y la Brevity Penalitation (BP) es una penalización por brevedad de la frase candidata, que se define como:

$$BP = \begin{cases} 1 & si \quad c > r \\ \\ e^{1-r/c} & si \quad c \leq r \end{cases}$$

donde c es la longitud de la frase candidata y r es la longitud de la frase referencia. De este modo, BLEU es un valor que oscila entre 0 y 1.

- **NIST:** NIST es una métrica muy similar a BLEU, ya que de hecho en su creación es tomada como referencia [53]. Como principal problema de asignar en BLEU el mismo peso a todos P_n es que se le asigna la misma importancia a todos los ngramas. En NIST lo que se hace es asignar más importancia aquellos n-gramas $(w_1,...,w_n)$ que surjan a partir de n-1-gramas $(w_1,...,w_{n-1})$ no únicos de tal forma que la palabra que se concatena a este último grama no tenga solo 1 posibilidad y por lo tanto dicha última palabras (w_n) proporcione información. Lo informativo que es cada grama se calcula de la siguiente manera:

$$info(w_1,...,w_n) = \log_2 \left(\frac{\#\ de\ apariciones\ del\ grama\ (w_1,...,w_n)}{\#\ de\ apariciones\ del\ grama\ (w_1,...,w_{n-1})} \right)$$

Al igual que BLEU, NIST trabaja con una media y una penalización condicionada por la comparativa del tamaño de la frase generada con respecto del de la referencia. Así NIST se define como:

$$NIST = \sum_{n=1}^{N} \left[\sum_{\substack{(w_1,...,w_n) \\ comun}} info(w_1,...,w_n) \Big/ ngramas\ candidata \right] \exp\left\{ \beta \log^2 \left[\text{mín}\left(\frac{c}{r}, 1 \right) \right] \right\}$$

- **METEOR:** Se presenta como una alternativa a las anteriores que trabaja a nivel de palabra dejando prácticamente de lado el resto de n-gramas [54]. Lo que se hace en este caso es hacer un mapeado de unigramas y ver el solapamiento de estos entre la frase generada y la referencia. Así, sean m el número de palabras coincidentes, t el total de palabras en la frase predicha y r el total en la frase referencia, se define la precisión como $P = m/t$ y el recall $R = m/r$. Con estos valores definidos, METEOR toma el valor de una media armónica entre dichos valores. Dicha media depende de un parámetro α que se utiliza para darle más importancia a uno u otro valor:

$$F_{mean} = \frac{P \cdot R}{\alpha \cdot P + (1-\alpha) \cdot R}$$

Como trabaja a nivel de palabra, F_{mean} no tiene en cuenta el orden. Por ello, METEOR computa una penalización de la siguiente manera: se toma la secuencia de palabras que se repiten en ambas frases y se divide dicha secuencia en el menor número de chunks posibles (ch), donde cada chunk está formado por palabras que se encuentran en el mismo orden en las dos frases. Con esto pasamos a definir la penalización:

$$Pen = \gamma \cdot \left(\frac{ch}{m}\right)^{\beta}$$

donde γ establece un máximo de penalización y β establece la relación funcional entre la fragmentación ($frag = ch/m$) y la penalización. Con todo esto, METEOR se define como sigue:

$$METEOR = (1 - Pen) \cdot F_{mean}$$

Los valores habituales para los parámetros de METEOR suelen ser $\alpha = 0.9$ (con el que damos 9 veces más importancia al recall que la precisión), $\beta = 3.0$ y $\gamma = 0.5$

- **ROUGE-L:** Es la primera de las métricas que no tiene su origen en la traducción automática, sino que lo hace de la generación de resúmenes [55]. ROUGE se asemeja a un recall donde se trabaja con distintos tipos de n-gramas y no solo con $n = 1$ como en METEOR. Asi, ROUGE-N, se define como el recall del caso anterior pero donde dependiendo del valor de N se trabaja con palabras (N=1), bigramas (N=2) y gramas de mayor orden. En el caso de ROUGE-L, la idea es la misma pero sin trabajar con un N fijo sino mediante un algoritmo obtener el tamaño máximo de n-gramas que presentan elementos comunes entre ambas sentencias y utilizar ese valor.

- **Slot ERROR (ERR):** Se trata de una métrica en la que lo que se analiza es cuáles de los slots presentes en el DA aparecen en la frase predicha. Se aleja del resto de métricas explicadas ya que no se basa en ninguna comparativa con una frase de referencia. Su definición es la siguiente:

$$ERR = \frac{M + S}{T}$$

donde T es el número de slots totales en todos los DAs, M el número total de slots que apareciendo en el DA no aparecen en la frase correspondiente y S es el número de slots totales que aparecen en la frase sin haber sido dados como información en el DA.

2.8.2. Evaluación humana

A su vez, aunque se puede hacer uso de métricas automáticas en el proceso de evaluación, es muy habitual encontrar evaluaciones humanas del sistema. Esta evaluación no solo se remite a la opinión de los usuarios finales dentro de un sistema real, sino que de forma intrínseca también se utilizan criterios humanos. Dentro de dichas evaluaciones hay posibilidad tanto de realizar solo una comparativa entre modelos en los que el usuario establece cual es mejor o mediante la evaluación absoluta en base a algún criterio. Los criterios más habituales son:

- **Corrección/estilo:** Evalúa la correctitud gramatical de la frase.

- **Naturalidad:** Determina si la frase generada se asemeja a la que podría haber generado un humano.

- **Adecuación/relevancia:** Mide lo adecuada que es una frase con el DA de partida.

- **Variabilidad:** Establece la capacidad del sistema para generar diferentes respuestas.

Es posible que las medidas automáticas y las evaluaciones humanas no coincidan, lo que vuelve a reflejar la dificultad de la evaluación. Esto también obliga a realizar una interpretación de los resultados obtenidos ya que los valores absolutos no derivan en una interpretación clara.

Capítulo 3

Desarrollo experimental

En este capítulo se explica todo el desarrollo experimental llevado a lo largo del trabajo. Dicho desarrollo implica dos fases: la obtención de una base de datos que permita desarrollar un NLG aplicado a EMPATHIC y la construcción de dos tipos de generadores para el EMPATHIC-NLG. En el proceso de obtención de datos se detallará el proceso de obtención de los datos y su etiquetado. En la segunda fase se detallarán los dos generadores diseñados para este trabajo: un sistema basado en plantillas al que se ha llamado GROWsetta y un generador estadístico denominado TGen.

Todo el desarrollo del EMPATHIC-NLG parte de la premisa de que el SDS desarrollado para EMPATHIC es un sistema de dominio especifico[56]. Con dominio especifico se quiere decir que los temas principales de los que se podrá conversar con el EMPATHIC-VC están muy ligados a los objetivos buscados en el proyecto. Así, el proyecto se ha desarrollado en base a cuatro escenarios sobre los cuales desarrollar un estilo de vida saludable para los mayores de 65: ejercicio físico, nutrición, tiempo libre y relaciones sociales. SDS de dominio especifico han sido desarrollado en muchos sistemas proveedores de información para, por ejemplo, el tiempo[57],servicios turísticos[58] o vuelos[59]. Sin embargo, por su finalidad y temática el modelo diseñado en este trabajo está más cercano a los servicios de asistencia médica y las aplicaciones de asistente personal para personas mayores[60] que a los mencionados anteriormente.

3.1. El modelo GROW

Una de las singularidades de este VC es que se ha desarrollado con la idea de hacer las labores de un coach profesional, lo que ha derivado en una estructura de diálogo específica y con una serie de patrones en lo que se refiere a los turnos del sistema. Los diálogos seguidos por un/a coach profesional siguen una estructura lógica de preguntas y respuestas con el objetivo de entender las necesidades, limitaciones y objetivos del usuario. Dichos objetivos deben ser definidos y aceptados por el propio usuario. Así, las tareas por parte del coach son promover la concienciación y responsabilidad del usuario y proporcionar una guía hacia unos objetivos realistas y saludables. Para poder seguir una estructura similar en el EMPATHIC-VC se ha decidido utilizar uno de los modelos de coaching más utilizados: el modelo GROW[61].

El modelo GROW, difundido por John Whitmore y creado por Graham Alexander en los años 80, facilita esquematizar las sesiones de coaching de una manera sencilla y clara. La gran ventaja de este modelo es que define una estructura de conversación con una secuencia de cuatro pasos fácil de seguir (figura 3.1). En este sentido, el nombre del modelo es un acrónimo de las cuatro fases del proceso del coaching.

- **La fase G o Goal** es principalmente una fase en la que se trata de explorar los posibles objetivos que el usuario pueda tener para alcanzar un estilo de vida más saludable. Se busca que

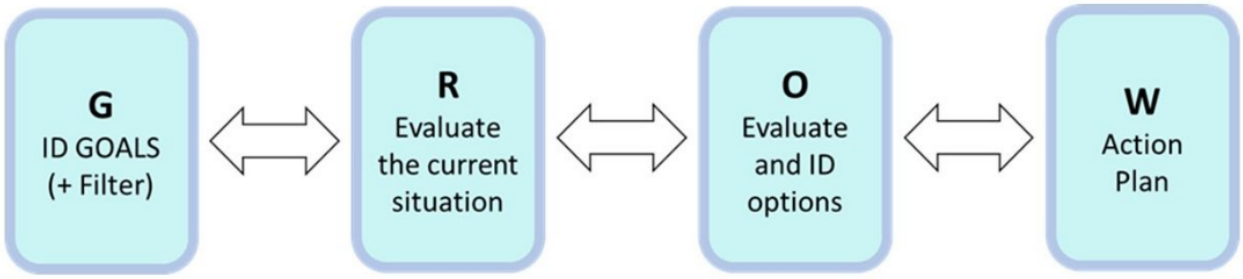

Figura 3.1: Secuencia del modelo GROW

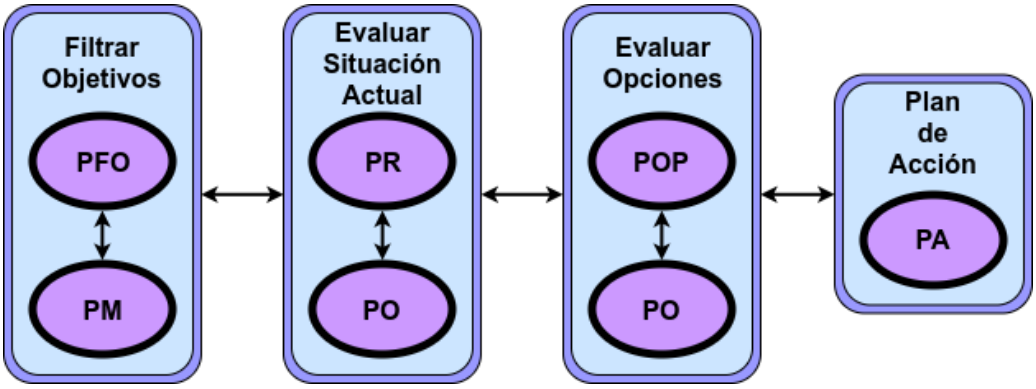

Figura 3.2: Uso de preguntas dentro del modelo GROW

sea el propio usuario quien los defina. Además, en esta fase se trata de motivar al usuario mediante preguntas.

- **La fase R o Realidad/Recursos** es una fase donde se trata de evaluar la realidad del usuario viendo lo alejado que se encuentra del objetivo definido en la sesión anterior. También se intenta buscar qué recursos tiene el usuario para reducir esa distancia además de identificar posibles obstáculos.

- **La fase O u Opciones** sirve para ver qué distintas acciones puede llevar a cabo el usuario para alcanzar el objetivo, haciendo una evaluación previa de ellas para ver cuál sería la más adecuada.

- **La fase W o Will** es la fase en la que el usuario elige una de las opciones del paso anterior de forma definitiva y establece el plan de acción que va a seguir para alcanzar el objetivo.

Al igual que en otras estrategias de coaching, la actitud del coach de dejar llevar el peso de la conversación al usuario hace que en la mayoría de sus turnos el coach haga uso sólo de preguntas. En el caso del modelo GROW, los tipos de preguntas y en que fase aparecen también están predeterminadas (figura 3.2). Se trabaja con ocho tipos de preguntas:

1. **Pregunta de Filtrado de Objetivo (PFO):** Pregunta utilizada para definir el objetivo o los subobjetivos del usuario. Con subobjetivos se refiere a objetivos más a corto plazo antes de alcanzar ese objetivo final, como si fueran pasos intermedios.

2. **Pregunta de Motivación (PM):** Pregunta que normalmente aparece en el contexto de la búsqueda del objetivo y que sirve para buscar una motivación personal del usuario que le ayude a alcanzar dicho objetivo. Normalmente son preguntas relacionadas con los beneficios que le daría alcanzar dicho objetivo.

E-Bienvenido Jorge ¿En qué te puedo ayudar? (PFO)

J-(Dubitativo y cabizbajo). Estoy bastante disgustado porque no estoy de acuerdo con mi forma de comer y me tiene preocupado

E-¿Y por dónde te gustaría empezar a cambiar tu forma de comer para alcanzar tu objetivo?(PFO)

J_ Creo que por comer cantidades más regulares en cada comida, como más repartido entre las comidas principales,

E-Discúlpame, Jorge, que insista.¿Y comer de forma regular en las comidas principales...? ¿Qué es para tí eso?(PFO)

J- No sé...

E- Sí sabes*... (silencio)*

J_ Vale, sí quizás tiene que ver con comer la misma ó aproximadamente la misma cantidad de comida en las tres comidas principales

E- ¿Comidas principales?(PR)

J- Sí, me refiero a desayuno, comida y cena

E- ¿Qué sucede con tu forma de comer? (PR)

J- Soy muy desordenado comiendo

E- ¿A qué te refieres con desordenado comiendo?/ ó ¿Desordenado comiendo?(PR)

J- Me refiero a que en algunos momentos del día como todo lo que se me pone por delante, y en otros apenas un trozo de cualquier cosa que pille en la nevera...

Figura 3.3: Ejemplo de diálogo inventado por el coach profesional

3. **Pregunta para testear Realidad/Recursos (PR):** Pregunta para definir el estado actual del usuario respecto al objetivo y los recursos de los que dispone para lograrlo.

4. **Pregunta para testear Obstáculos (PO):** Pregunta para determinar los obstáculos que tiene o que puedan aparecer en el camino hacia objetivo del usuario.

5. **Pregunta para generar Opciones (POP):** Pregunta cuya idea principal es extraer todas las posibilidades en forma de acciones que el usuario tiene para alcanzar el objetivo.

6. **Pregunta para establecer plan de Acción (PA):** Pregunta con la que se busca que el usuario elija de forma definitiva una de las acciones a realizar como plan a seguir para alcanzar su objetivo.

7. **Pregunta de Seguimiento (PS):** Pregunta utilizada para saber cómo le ha ido al usuario con los planes de acción definidos en sesiones anteriores.

8. **Pregunta de Alerta (PAL):** Pregunta que sirve para conocer el estado de salud del usuario con idea de adaptar las posibles acciones que pueda llevar a cabo.

3.2. Adquisición de datos

La adquisición de los datos utilizados para la implementación del NLG se divide en dos partes. La primera parte no implica a los usuarios finales y ha sido suministrada en su gran mayoría por una coach profesional, mientras que la segunda parte se ha extraído de conversaciones con usuarios reales. De esta forma, se les está implicando desde el principio, como era objetivo del proyecto.

Inicialmente, para entender la forma de interactuar de un coach en una sesión con personas reales y saber qué se podía llegar a implementar, se consultó a una coach profesional. Dicha profesional aparte de suministrar unas pautas y hacer entender cuál debía ser la labor del asistente, elaboró unos diálogos inventados (figura 3.3) de lo que podía ser una sesión con distintos usuarios, en distintas fases del proceso de coach (varias sesiones diferentes con el mismo usuario) y

Figura 3.4: Interfaz gráfica para el usuario en la plataforma WoZ [1]

tratando los cuatro escenarios que EMPATHIC ha definido como temas a tratar. Además, dos personas del propio proyecto grabaron dos sesiones con ella a modo de simulación de lo que sería una conversación real de coaching. Todos los turnos del coach de dichas sesiones y de las conversaciones inventadas han sido incluidos como parte de la base de datos a utilizar. De hecho, en los inicios de la implementación del primer NLG los únicos datos con los que se han contado han sido estos que se mencionan hasta aquí.

A la hora de involucrar a los usuarios para la adquisición de datos se debía hacer mediante una metodología que permitiese capturar datos de todo tipo para el proyecto y que permitiese ver cómo es la interacción de una persona mayor de 65 años con un sistema del tipo que queríamos implementar. De este modo, no sólo se ha querido extraer datos sino que también se han analizado las preferencias de los usuarios frente al VC. Es por ello, que el sistema utilizado para estas sesiones se ha desarrollado con un aspecto igual o prácticamente igual al que se va a utilizar posteriormente a lo largo de todo el proyecto EMPATHIC. Ante estas necesidades, se ha utilizado una plataforma Wizard of Oz (WoZ) para esta primera interacción con usuarios reales [1].

Las herramientas WoZ son muy utilizadas dentro del mundo de la investigación y no dejan de serlo en el mundo de los SDS [62, 63, 64]. La idea de estos sistemas es hacer creer al usuario que está interaccionando con una herramienta virtual pero en realidad el encargado de llevar a cabo las sesiones es una persona, que denominaremos Wizard, que remotamente hace la función de la herramienta. En el caso de EMPATHIC, se ha utilizado una de las pocas plataformas de WoZ que es de uso abierto. Dicha herramienta fue desarrollada por Schlögl et al. [65], es la base de un nuevo marco de trabajo en Internet para los WoZ y permite la integración las tecnologías del lenguaje, entre otras cosas. Aprovechando esto, en EMPATHIC se ha integrado un reconocedor automático del habla para captar las respuestas del usuario, un sistema de síntesis de voz para generar las respuestas del Wizard , un avatar sencillo para la interfaz de usuario (figura 3.4) y un sistema de grabación que permitía ver al usuario en la interfaz del Wizard (figura 3.5). Con todo esto, la función que tenía el Wizard era la de hacer de coach dando respuestas a los usuarios. Para realizarla, como se puede ver en la figura 3.5, el Wizard cuenta con una serie de frases predefinidas para hablar con el usuario así como la capacidad de escribir cualquier tipo de repuesta con el teclado. De todo esto, para la base de datos del NLG lo que queda es un archivo que la herramienta de WoZ genera automáticamente con todas las intervenciones del coach y del usuario durante dichas conversaciones (las del coach son aquellas que al principio de línea tienen una S en la

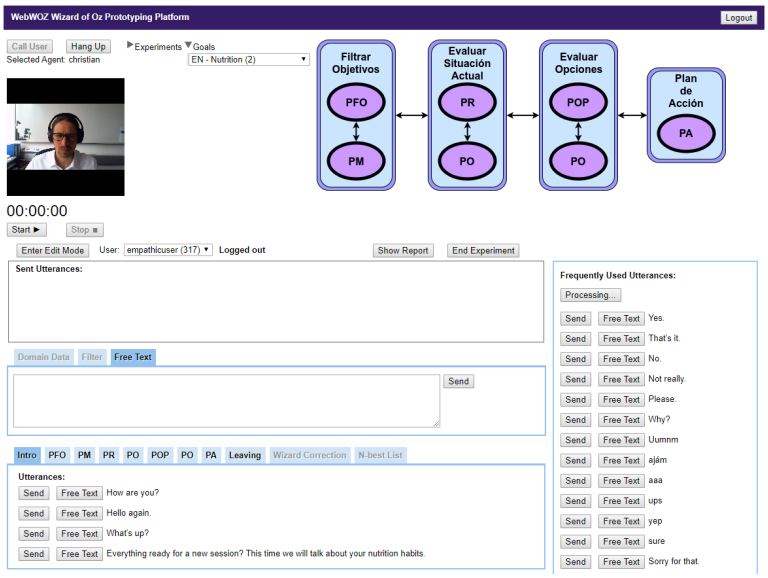

Figura 3.5: Interfaz gráfica para el Wizard en la plataforma WoZ [1]

S 00:02:00.48 00:02:05.72 ¿Podrías concretar en pocas palabras tu objetivo?
U 00:02:09.70 00:02:12.49 Creo que comiendo menos, se puede estar mejor.
S 00:02:17.17 00:02:21.30 Imagina que ya lo has conseguido, ¿cómo te ves?
U 00:02:24.27 00:02:26.00 Ah, verme me veo siempre bien.
S 00:02:33.56 00:02:39.67 Muy interesante. Fantástico. ¿Qué beneficios obtendrías si consiguieras comer de forma más equilibrada?
U 00:02:42.99 00:02:48.45 Quizá algo más... algo menos de peso y una mejor forma física.
S 00:02:50.24 00:02:53.77 Eso es. ¿Por qué?

Figura 3.6: Fragmento de una conversación entre el VC y un usuario real

figura 3.6).

Las sesiones se han hecho con participantes de los 3 países (tabla 3.1) y siguiendo una metodología de adquisición de datos que incluye dos sesiones por usuario (figura 3.7). Ambas son de una duración cercana a los 10 minutos. En la primera sesión se realiza una sesión en la que el coach realiza preguntas al usuario para conocerlo y hablar sobre sus gustos y aficiones, de tal modo que el usuario coja confianza en la interacción con una maquina hablando de temas amenos. Tras dicha sesión, se da descanso al participante durante 15-20 minutos. En la segunda, el escenario tratado es el de nutrición y el objetivo definido para los Wizards de los diferentes países es que al menos sean capaces de completar la etapa Goal del modelo GROW y si es posible por tiempo ir más allá a partir de ese objetivo definido. De este modo, la previsión de datos obtenidos mediante este formato de adquisición es todas aquellas intervenciones que se den en las 360 sesiones (180 de introducción y 180 de nutrición) de 10 minutos, no todas en el mismo idioma.

España	**90 participantes**
Noruega	**60 participantes**
Francia	**30 participantes**

Tabla 3.1: Número de participantes por país

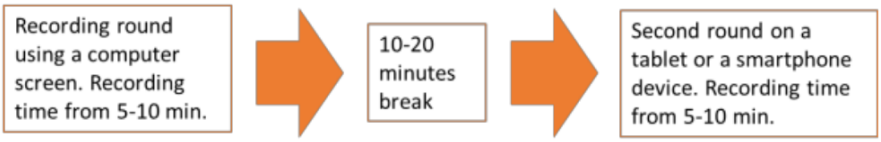

Figura 3.7: Procedimiento para la grabación de las sesiones iniciales con usuarios

3.3. Diseño de la anotación

A la hora de establecer el sistema de etiquetas para la anotación de los actos de diálogo en EMPATHIC, desde un principio se tuvo una idea clara: la utilización de etiquetas ligadas al modelo GROW y otras asociadas a elementos lingüísticos de forma más general. Otra de las decisiones que se tomó prácticamente desde el principio es la de seccionar el diálogo en su unidad más pequeña de información, las sentencias.

Las etiquetas asociadas al modelo GROW son básicamente los 8 tipos de preguntas en los que se basa el modelo GROW (ver sección 3.1), mientras que para definir el otro conjunto de etiquetas se han tomado como referencia los dos modelos de anotación síncronos explicados en la sección 2.7.1. Es por ello que los elementos lingüísticos que entran en juego a la hora de definir las etiquetas del segundo conjunto son funciones comunicativas. Partiendo de considerar por validas todas las funciones comunicativas/etiquetas que proporcionaban tanto SWBD-DAMSL y DIT++, se han escogido aquellas con un mínimo de aparición dentro del primer bloque de datos a anotar, de forma muy similar a lo que se hizo para construir SWBD-DAMSL a partir de DAMSL. Dicho primer bloque de datos se refiere a aquel primer conjunto que fue suministrado en su gran mayoría por la coach profesional.

El problema es que con conjunto de etiquetas GROW no tenía sentido etiquetar frases fuera de la temática coach. Así, se plantearon dos posibles formas de abordar la anotación: asignar dos etiquetas (la etiqueta GROW y la basada en conceptos lingüísticos) a aquellas sentencias dentro del modelo GROW y solo una con el segundo tipo al resto de sentencias o dejar todas las sentencias con una etiqueta eliminando la parte lingüística de las frases relacionadas con el modelo de coaching. Finalmente, se vio que agregaba poca información añadir una segunda etiqueta a las sentencias dentro del modelo GROW ya que casi en su totalidad las frases eran preguntas. De este modo, se ha diseñó un sistema de anotación unidimensional con una única etiqueta por sentencia, una del modelo GROW para aquellos sentencias de dominio especifico y una basada en su función comunicativa para el resto de frases.

La tarea de NLG destaca por la necesidad de unos actos de diálogo más específicos que otras herramientas basadas en la anotación de los DAs. Es por ello que es muy habitual que los DAs de los NLG en su formato incluyan atributos o slots [66, 40, 28]. Los atributos son elementos dentro de una frase que podrían ser reemplazables por otros semejantes a ellos. Por ejemplo, algunos de los atributos podrían ser nombres de personas o comida como en la siguiente frase: *"José, ¿quieres aumentar la cantidad de uvas?"*, donde *<user_name>* es *José* y *<food>* toma el valor de *uvas* pero dichos valores podrían ser sustituidos por otro nombre al inicio de la frase y otra comida al final y seguiría siendo una frase con sentido.

La anotación de estos atributos está fuertemente ligada al concepto de plantilla, ya que si en la frase en vez del valor del atributo se pone el nombre del atributo se pasa a tener un plantilla. En el ejemplo anterior la plantilla sería *"<user_name>, ¿quieres aumentar la cantidad de <food>?"* con *<user_name>* sustituible con cualquier nombre propio de persona y *<food>* con cualquier alimen-

Entidad	Ejemplos
Personas	Juan, Carmen, Isabel...
Nombre usuario	Juan, Carmen, Isabel...
Nombre del coach	Juan, Carmen, Isabel...
Familiares/otras personas	mi marido, un compañero de trabajo, mi cuidador...
Acciones en infinitivo	comer, salir, pasear...
Tema de conversación	hábitos de nutrición, viajes, ejercicio físico...
Aficiones y deportes	tenis, baile...
Alimentos	cereales, fruta...
Grupos y obras musicales	The Ramones, The Dark Side of the Moon...
Libros	El Señor de los Anillos, Harry Potter...
Películas y series	El golpe, Birdman, El Padrino...
Cuadros, esculturas y otros artes	El Grito, La Mona Lisa...
Números cardinales	5, sesenta...
Números ordinales	primero, decimocuarto, 8º...
Cantidades de tiempo	un año, tres meses...
Cantidades	poca, muchísimo...
Enfermedades y patologías	lumbago, hipertensión...
Emociones	pena, alegría...
Fechas	el lunes, el 24 de mayo, pasado mañana...
Frecuencias	cada 3 días, habitualmente,...
Lugares y edificios/organizaciones	hospital, Calle Rosendo, oficina de correos...
Meteorología	llueve, sol radiante...
Nacionalidades	china, franceses...
Objetos de utilidad	tijeras, motocicleta, lata...

Tabla 3.2: Entidades a anotar

to. La importancia de extraer de una frase su plantilla es que aumenta la variabilidad del conjunto de datos ya que de un único plantilla se pueden sacar infinitas frases si no se limitan los valores de los atributos. Además, sabiendo que uno de los sistemas a generar, GROWsetta, iba a ser basado en plantillas aumentaba la lógica de realizar dicha extracción.

Los atributos escogidos para su posible aparición en las frases son los que se muestran en la tabla 3.2. A la anotación de dichos atributos se le conoce como detección de entidades en concordancia con como se denomina a este proceso en la anotación de los turnos de usuario, muy importante para el diseño del NLU. De hecho, las entidades que se pueden incluir para el NLG son aquéllas que dicho módulo pueda detectar y es por ello que las entidades escogidas, salvo ligeras modificaciones, son las mismas que en el caso de la anotación semántica, que es como se llama al etiquetado de los turnos del usuario.

Hay algunas de las entidades recogidas en la tabla que con los ejemplos son autoexplicativas. Sin embargo, algunas de ellas deben ser explicadas para evitar confusiones.

- **Personas, Nombre usuario y Nombre del coach** son nombres propios de persona a diferencia de **Familiares/otras personas** donde se hace referencia a personas sin usar su nombre propio. La diferencia entre las tres primeras entidades que están asociadas a nombre propios de persona es que **Nombre usuario** y **Nombre del coach** solo se utilizan en los casos específicos del nombres del usuario y del coach virtual, respectivamente, mientras que **Personas** quedan para el resto de nombre propios que se hayan dicho en la conversación.

- **Acciones en infinitivo** es fácil de entender que son aquellas acciones que no aparecen conjugadas. El hecho de solo interesar dicho tipo de acciones es porque solo esas son directamente reemplazables en un plantilla, lo cual es importante más adelante en el diseño de los generadores. Para ejemplificar, si el usuario dice *"Yo suelo correr"* y en la entrada del usuario se detecta *correr* como una entidad se puede utilizar directamente para reemplazar en algún plantilla adecuado. En cambio, si dice *"Yo corro"* aunque se detecte *corro* como una entidad de *acción* no es directamente reemplazable en una frase donde el sistema quiere hablar sobre que el usuario corre.

- **Tema de conversación** no se utiliza para establecer el tema de la conversación de la sentencia, sino para identificar si el coach menciona el tema de conversación o el posible tema de conversación de manera clara. Un ejemplo de esto sería *"¿Te gustaría hablar sobre nutrición?"* donde **Tema de conversación** tendría como identidad *nutrición*.

Aún con la inclusión de los atributos, se seguía detectando que los actos de diálogo eran poco específicos para la tarea de la implementación de un NLG. Es por ello que se decidió trabajar con una taxonomía de etiquetas y subetiquetas. Dicha taxonomía se ha dado por definida y cerrada con la llegada de las conversaciones de los usuarios reales que han dejado ver que las sesiones de introducción con el usuario (la primera sesión de las dos que se hacían) también tenían una estructura muy repetitiva pero diferente de las basadas en GROW. Por ello, a aquellas intervenciones habituales dentro de la primera sesión se les ha asignado un bloque independiente dentro de dicha taxonomía.

Con todo esto, la anotación de los actos de diálogos para EMPATHIC presenta una estructura final con seis elementos a asignar en cada sentencia: etiqueta principal, subetiqueta, entidades y tres atributos no presentes explícitamente en las sentencias pero sí fácilmente extraíbles: géneros de usuario y sistema y la polaridad. Así el formato de un DA sería:

$$\begin{aligned} &label\&sublabel(<attribute\_1>=value\_1,...,<attribute\_n>=value\_n,\\ &<user\_genre>=M/F/I,<system\_genre>=M/F/I,<polarity>=P/N)^{*} \end{aligned} \tag{3.1}$$

Comenzando con las etiquetas principales, son 10 etiquetas las que se han considerado. Las ocho primeras son las ocho preguntas del modelo GROW. Luego, se tiene la etiqueta *Int* que recoge todas aquellas sentencias que están dentro de la estructura habitual de una primera sesión del coach con el usuario. Y por último, para aquellas sentencias de carácter más general que se pueden usar tanto en el transcurso de una sesión GROW como de una sesión de inicio tenemos la etiqueta *Gen*. Cada etiqueta principal tiene su conjunto de subetiquetas asociadas como se presenta posteriormente. En resumen, el conjunto de etiquetas principales es el siguiente:

- **Pregunta de Filtrado de Objetivo (PFO)**
- **Pregunta de Motivación (PM)**
- **Pregunta para testear Realidad/Recursos (PR)**
- **Pregunta para testear Obstáculos (PO)**
- **Pregunta para generar Opciones (POP)**
- **Pregunta para establecer plan de Acción (PA)**
- **Pregunta de Seguimiento (PS)**
- **Pregunta de Alerta (PAL)**
- **Introducción (Int)**
- **General (Gen)**

En cuanto a las subetiquetas,se distinguen tres grupos: las asociadas a las etiquetas del modelo GROW, las de la fase de introducción y las generales basadas en funciones comunicativas. A la hora de seleccionar las subetiquetas del modelo GROW, éstas han estado basadas en el contenido de

Figura 3.8: Taxonomía de etiquetas asociadas al modelo GROW

las preguntas en las distintas fases. Como se puede apreciar en la figura 3.8 algunas de las subetiquetas pueden aparecer bajo el uso de diferentes etiquetas pero también existen algunas que son exclusivas de una sola etiqueta principal.

En el caso de las subetiquetas de Introducción, se definen otra vez en base al contenido de las frases que aparecen con frecuencia en una primera sesión. Dichas primeras sesiones se definieron con unos objetivos diferentes a los de las sesiones de coaching. Por una parte, se intenta que el usuario se habitúe al trato o, por lo menos, tenga un primer contacto con lo que supone una interacción con un VC. Para ello, la conversación se centra a priori en temas en los que el usuario se puede sentir más a gusto. Por otra parte, se recogen los primeros datos del usuario y se le da cierta información de todo lo que engloba a las conversaciones que van a mantener a partir de dicha sesión. En resumen, los temas de conversación de dicha sesión serían:

- Conocer al usuario a nivel personal (nombre, edad, otros datos personales,...)
- Explicar en que consiste el coaching, el sistema implementado y el proyecto EMPATHIC.
- Hablar sobre los hobbies del usuario, teniendo como temas habituales: los viajes y la música.

En este caso, como se ha explicado antes para todas estas frases la etiqueta es *Int* mientras el conjunto de subetiquetas es el que sigue:

1. **¿Tu nombre?**
2. **Deletrea nombre**
3. **¿He pronunciado bien?**
4. **Presentación máquina**
5. **Petición de paciencia**
6. **Información sobre sistema**
7. **Información sobre EMPATHIC**
8. **¿Conoces coaching?**
9. **Información sobre coaching**
10. **¿Por qué vienes?**
11. **Pregunta sobre viajes**
12. **Comentario sobre viajes**
13. **Pregunta sobre música**
14. **Comentario sobre música**
15. **Pregunta sobre otros hobbies**
16. **Comentario sobre otros hobbies**
17. **Preguntas personales**
18. **Pedir anécdota**

Por último, las subetiquetas recogidas bajo la etiqueta principal *Gen* son aquellas etiquetas que se ha explicado antes que se han escogido partiendo de los sistemas de etiquetas SWBD-DAMSL y DIT++. Se trata de un conjunto de etiquetas definidas a partir de su función comunicativa y que se utilizan dentro de este sistema de etiquetas para anotar todas aquellas expresiones que se escapan de las estructuras de las sesiones de inicio y de coach pero que sí son de importancia en el desarrollo de la conversación. Para aquellas expresiones sin contenido semántico de importancia o no interpretables, la anotación permite la anotación de elementos *No clasificable*, pero en ese caso dichas sentencias no se incluyen en el entrenamiento de los modelos de los generadores. Siguiendo con las subetiquetas de dominio abierto, se ha decidido seguir algunas pautas en cuanto a jerarquía vistas en DIT++, como se aprecia a continuación. Todas las subetiquetas se agrupan bajo tres subconjuntos (figura 3.9) que tienen una fuerte conexión con el sistema de etiquetas desarrollado por Bunt [43].

- **Informar**, que es un equivalente de *Information Providing Functions* de DIT++, es decir, son aquellas funciones que le suministran una información al emisor. En este caso, también incluye algunas de las etiquetas que DIT++ incluía dentro de las *Action Discussion Functions*.

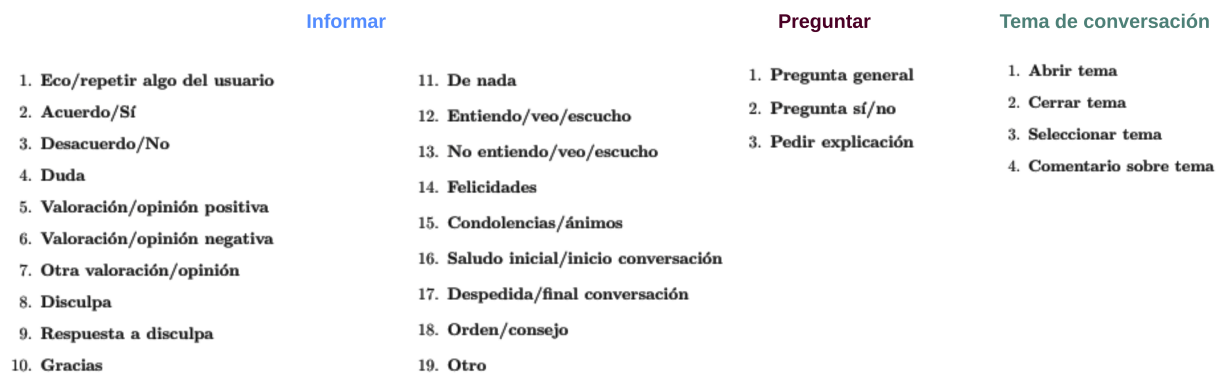

Figura 3.9: Subetiquetas del conjunto general

- **Preguntar** se establece como equivalente a las *Information Seeking Functions*, que era el otro subconjunto dentro de *Information Transfer Functions* en DIT++, y sirve como etiqueta para todas aquellas preguntas fuera del contexto de los dos anteriores conjuntos de etiquetas (GROW e Int).

- **Tema de conversación** se ha escogido también tomando como referencia una de las 10 dimensiones de DIT++ (*Dialogue Strucuture Management*). Bajo este nivel se etiquetan todas aquellas intervenciones en las que el coach trata de definir la temática del diálogo.

De este modo, se tiene una taxonomía con 10 etiquetas principales y más de 100 subetiquetas.

Fuera de la taxonomía de etiquetas-subetiquetas, se tendrían las entidades que se añaden al DA en forma de atributos, si es que la frase contiene alguna, y los tres atributos implícitos que aparecen de forma obligatorio en cada DA. Estos tres atributos aparecen con la idea de limitar el uso de ciertas frases a ciertos contextos. La primera sería restringir frases que den a entender el género del usuario o del coach a los casos en el que el género del participante y el coach coincidan con el asignado en el acto diálogo. Para ello, para todas las sentencias se da la posibilidad de anotar ambos géneros por separado con *Masculino* o *Femenino* en caso de que a través de dicha sentencia se detecte el género de uno de ellos. En caso de que el género no quede reflejado, se da la posibilidad de etiquetar el género como *No identificable*. Hay que destacar que para el noruego debido a las características del idioma no tiene sentido dicha anotación, ya que, como sucede también en el inglés, en las frases no se puede detectar el género de los participantes. Por su parte, la polaridad puede solo tomar los valores de *Positivo* o *Neutro*, ya que se entiende desde el proyecto que la polaridad que muestra el sistema no puede ser negativa. De este modo, se pretende hacer una división de los datos en función de con qué polaridad puedan ser expresados.

3.4. Proceso de anotación

Una vez ha quedado establecido el conjunto de etiquetas y demás elementos que componen el acto de diálogo, se sigue con la explicación de los dos procedimientos para etiquetar la base de datos. La base de datos queda dividida en la parte suministrada por la coach profesional y la que se obtiene de las conversaciones con usuarios reales.

Inicialmente, los datos que se tenían para la tarea eran muy limitados, ya que no se tenían todavía los datos de las conversaciones reales. Sin embargo, con dichos datos se necesitaba iniciar tanto la definición de las etiquetas antes explicadas como el primer prototipo del NLG. Para dicha

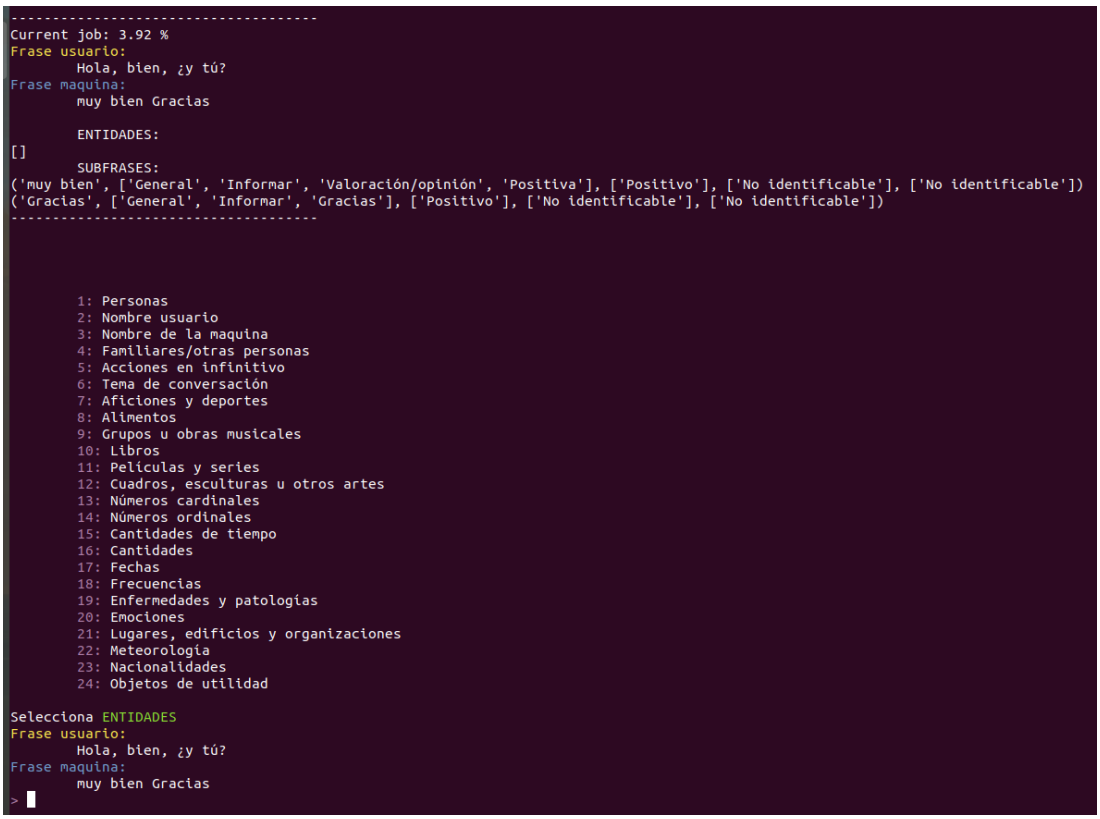

Figura 3.10: Plataforma de anotación de los actos de diálogo

parte, se decidió hacer un trabajo personal de etiquetado por dos motivos principales: entender y conocer a fondo de la base de datos de la que se partía y el hecho de que el sistema de etiquetado estaba en proceso de definición. Además, la base de datos era lo suficientemente reducida (2481 sentencias) para no ser un trabajo inabordable por una sola persona.

Con la llegada de los datos que involucraban a usuarios reales el sistema de etiquetas terminó por definirse. Dichos datos duplicaban los anteriormente anotados (5985 sentencias). Esto, unido a que en otras anotaciones del proyecto se estaban utilizando anotadores contratados para realizar los distintos etiquetados, derivó en que se tomara la decisión de realizar la segunda parte aprovechando dichos anotadores.

A estas personas se les proporcionó una guía de anotación, una visión general de las conversaciones y del proyecto y una plataforma de sencilla utilización para el proceso de anotación (figura 3.10). En dicha plataforma, seccionaban los turnos del sistema en sentencias para posteriormente asignarles los distintos elementos que constituyen el DA. Para facilitar la tarea y por ser necesario el contexto para un buen etiquetado, etiquetaban conversaciones completas, además de que para cada turno del coach contaban con el turno anterior del usuario por pantalla, como se aprecia en la figura 3.10.

3.5. GROWsetta

3.5.1. GROWsetta: el concepto

Para el primer NLG de EMPATHIC se decidió usar un sistema basado en plantillas. La principal razón de dicha elección es que la cantidad de datos con la que se contaba inicialmente no era suficiente para obtener con seguridad un sistema estadístico de calidad. Así que de este modo lo que

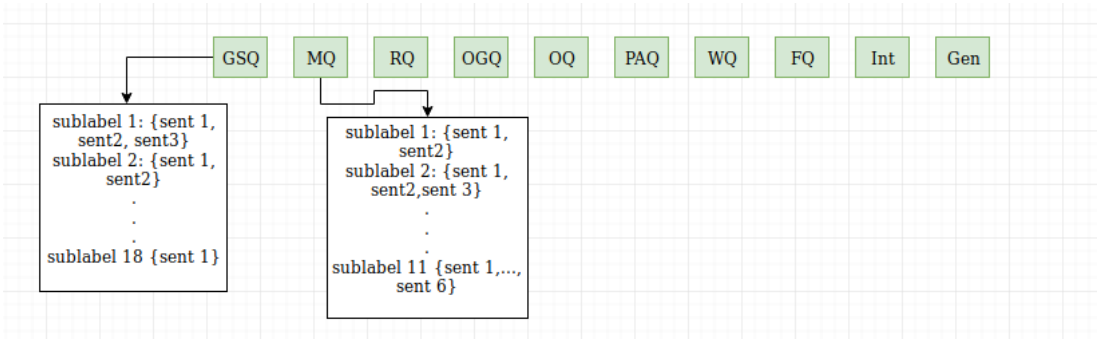

Figura 3.11: Estructura interna de GROWsetta

se asegura el sistema es reducir la probabilidad de errores lingüísticos en la formación de las frases a prácticamente cero . Además, se ha considerado que el sistema basado en plantillas era mucho más sencillo de implementar que un sistema basado en reglas gramaticales, teniendo en cuenta que se trabaja con un sistema multilingüe (dos de cuyas lenguas eran escasamente conocidos para el autor de este trabajo y desarrollador del NLG) y que tratándose de un sistema de dominio reducido el número de plantillas a incluir en el generador no iba a ser excesivamente grande. Una vez decidido el tipo de generador, se ha considerado implementar un NLG desde cero al que se ha denominado GROWsetta, por estar adaptado al modelo GROW [61] y por ser un sistema basado en plantillas al igual que lo es Rosetta[29], un generador alguna vez utilizado dentro del grupo de investigación.

La estructura interna de GROWsetta es la que se muestra en la figura 3.11. Dicha estructura está ligada a la selección de etiquetas y subetiquetas explicadas en la sección 3.3. Así, se tienen 10 bloques diferenciados, donde cada uno viene definido por cada una de las 10 etiquetas principales (cajas verdes de la figura 3.11) definidas para EMPATHIC: las 8 etiquetas principales del modelo GROW más las etiquetas asociadas a la sesión de introducción y al caso general. Dentro de cada bloque (cajas blancas) tendremos asignados a cada subetiqueta uno o varios plantillas de los que poder elegir posteriormente uno para formar la frase. El número de subetiquetas de cada bloque se corresponde con la cantidad de subetiquetas que se le ha asignado a cada etiqueta principal. A su vez, el número de plantillas asignados a cada pareja etiqueta-subetiqueta no es fijo, depende del número de frases que se haya considerado que cumplen con las características necesarias para integrarse dentro del modelo.

A la hora de seleccionar las plantillas, son tres las consideraciones que se han tenido en cuenta. La primera es una característica más asociada al generador que a las plantillas en sí: adaptabilidad al DM. Lo que se quiere decir con esto es que las plantillas que se han ido incluyendo en GROWsetta son aquellos que va a poder producir el gestor de diálogo. Incluir plantillas que el DM no va a ser capaz de producir sería realizar un trabajo en balde. Las otras dos características que se buscan con las plantillas elegidos (figura 3.12) son generalidad y variabilidad. Con la generalidad lo que se busca es que la frase no sea demasiado específica para un caso concreto, ya que plantillas demasiados concretos haría que el número de plantillas a incluir fuera muy alto y que el número de atributos para especificar la plantilla a elegir también lo fuera. En cuanto a la variabilidad, esta claramente viene reflejada por el número de huecos a rellenar que haya en la plantilla. Cuanto mayor es el número de huecos, más variabilidad da un plantilla. Como se ve en los ejemplos de la figura3.12, la primera plantilla permite preguntar sobre el objetivo que se quiere conseguir en el tema de nutrición, pero si en el DA se cambia $< topic >$ por cualquier otro tema la plantilla sigue siendo igualmente válido.

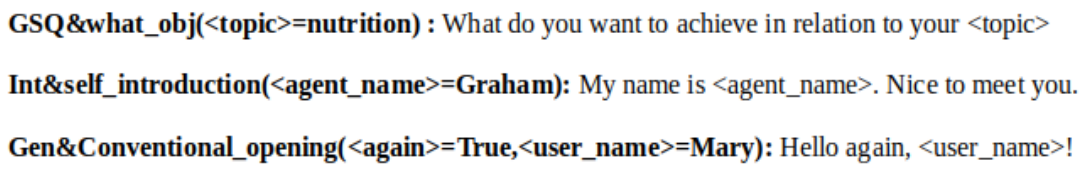

Figura 3.12: Ejemplos de DAs incluidos en GROWsetta

Con lo explicado hasta aquí, ya se tiene el mapeado típico de los sistemas de plantillas, donde a cada DA se le asignan una serie de plantillas. Se debe recordar que los DAs para EMPATHIC no solo consisten en etiqueta y subetiqueta sino que incluyen una serie de atributos que también influyen a la hora de seleccionar la plantilla adecuado. El número de atributos no es fijo aunque sí que se debe recordar que hay tres atributos obligatorios (géneros de usuario y sistema y polaridad). Dentro de los atributos se distinguen de dos tipos:

- **Atributos de selección:** Dichos atributos son aquellos que solo tienen una función, que es la de ayudar al generador a seleccionar un plantilla adaptado al contexto de la conversación, incluyendo en ello tanto el estado de la conversación como las características del usuario. Los tres ejemplos más claros de este tipo de atributo son los tres atributos obligatorios. Dichos atributos sirven para reducir el número de plantillas posibles a seleccionar, ya que puede haber frases que no se pueden decir con una polaridad determinada o que no se puedan utilizar porque el género de alguno de los participantes no concuerde con la realidad. Sin embargo, no son los únicos atributos de este tipo, ya que, por ejemplo, en el último ejemplo de la figura 3.12 aparece <again>, que es un atributo que viene a indicar si es la primera vez que se ha saludado al usuario o no. En caso de que no se le haya saludado antes, la frase que aparece en el ejemplo no tendría ningún sentido por el estado de la conversación. Por ello, para ese tipo de casos este tipo de atributos es necesario.

- **Atributos de remplazo:** La función principal de dichos atributos es la de asignar los valores que se van a sustituir en los huecos de la plantilla. Aun así, también tienen su importancia a la hora de escoger un determinado plantilla. De primeras, no se puede escoger un plantilla que tenga un hueco asociado a un atributo que no aparezca en el DA, ya que no se sabría con qué sustituir dicho hueco. Además, aquellas plantillas que contengan un hueco para cada uno de los atributos de remplazo se escogen con mayor probabilidad que aqula plantilla que no contenga ningún hueco o un número menor de huecos que el número de atributos de este tipo, es decir, cuanto más huecos haya dentro de la plantilla mayor probabilidad de que se escoja, siempre que los huecos sean asociados a atributos que aparezcan en el DA.

  Por otro lado, dentro de estos atributos de reemplazo hay una distinción entre ellos: los detectados como entidades y los predefinidos. En el primer tipo el DM toma los valores detectados por el NLU como entidades para dar el valor de los atributos. Dicho valor se reemplaza directamente en el hueco de la plantilla ya que viene dado en el idioma en el que se debe generar la frase. Además, este tipo de atributo hace un efecto espejo ya que se va a utilizar la misma expresión utilizada por el usuario para introducirlo en la frase. En los ejemplos de las plantillas, <*user_name*> y <*agent_name*> son atributos de este tipo. Su gran ventaja es que el número de valores a sustituir son ilimitados (limitados a la capacidad del NLU de detectar entidades).

  El segundo tipo hace referencia a una serie de atributos a los que ya se les tiene asignados unos posibles valores de sustitución cuando aparezcan. Hasta el momento solo dos tipos de atributos tienen valores predefinidos: <*action*> y <*topic*>. Esto no quiere decir que estos dos atributos no se puedan reemplazar por valores de las entidades del NLU, sino que tienen una

<action>=feel_alone			
Variante	**Español**	**Francés**	**Noruego**
action	no sentirse solo sentirse acompañado tener sensación de compañía	ne pas se sentir seul	ikke føle deg alene ikke føle deg ensom
action_inv	sentirse solo no sentirse acompañado no tener sensación de compañía	se sentir seul	føle deg alene føle deg ensom
action_inv_pr_si_2_s	te sientes solo; no te sientes acompañado; no tienes sensación de compañía	tu te sens seul	føler deg alene føler deg ensom
action_inv_pr_si_2_s_Q (only for Norwegian)			føler du deg alene føler du deg ensom
action_pr_si_2_s	no te sientes solo te sientes acompañado tienes sensación de compañía	tu ne te sens pas seul	Ikke føler deg alene ikke føler deg ensom
action_pr_si_2_s_Q (only for Norwegian)			føler du deg ikke alene føler du deg ikke ensom

Tabla 3.3: Variantes del atributo con reemplazo para el valor predefinido de *<action>=feel_alone*.

serie de valores adicionales predefinidos. Los atributos predefinidos para *<action>* son cada una de las acciones objetivo que se ha considerado habitual en las conversaciones en el marco EMPATHIC, mientras que en el caso de *<topic>* los valores predefinidos están asociados con los temas habituales de conversación en EMPATHIC. Todos estos valores los proporciona el DM en un formato independiente del idioma, es decir, es el mismo para castellano, francés y noruego. Posteriormente, GROWsetta se encarga de gestionar en qué idioma generar el texto a reemplazar a partir de dicho valor. La ventaja del caso de los predefinidos es que se controla totalmente los valores que se van a sustituir y permite hacer diferentes transformaciones a partir de ese valor. En el caso de *<action>* (tabla 3.3) permite conjugar el verbo e incluso utilizar la pauta no saludable del usuario (<action_inv>y sus variantes) en vez del verbo del objetivo en los verbos a sustituir en el hueco de la plantilla. De momento, todas estas cosas no se pueden hacer a partir de la detección de las entidades, ya que no se ha implementado nada para poder definir si el verbo es el objetivo o el problema ni el tiempo verbal en el que se presenta. Como se ve también en la tabla la idea es dar más de un reemplazo posible para cada variante en cada idioma por aumentar la variabilidad de las frases.

Como ya se ha comentado anteriormente el EMPATHIC-VC es multilingüe (español, francés y noruego), lo que implica que el NLG también lo sea. El hecho de trabajar con plantillas ha hecho que el proceso de adaptación del sistema de un idioma a otro sea un proceso de traducción. Más allá del proceso de traducción, el hecho de trabajar con tres idiomas hace que algunas consideraciones no tengan el mismo sentido en todos los idiomas. Por ejemplo, ya se ha visto (tabla 3.3) que para el noruego se encuentra una variante diferente en los modos verbales, ya que para las preguntas el sujeto (*du* es tú o usted en noruego) debe aparecer incrustado en medio del modo verbal como lo haría en inglés. Otra semejanza entre el noruego y el inglés y que lo diferencia de los otros dos idiomas es la indiferencia del idioma ante el género del emisor y receptor de la conversación (tabla 3.4). En los otros dos idiomas sí que es importante que se definan las frases que se pueden utilizar para cada combinación de géneros, mientras que en el noruego no se hace dicha anotación, ya que siempre es indiferente, por lo que dichos dos atributos obligatorios no lo son para el noruego.

Con todo esto y antes de pasar a detallar la implementación, el proceso que sigue GROWsetta

Idioma	**Coach hombre** **Usuario mujer**	**Coach mujer** **Usuario hombre**
Español	Estoy cansado y veo que tu también estas cansada.	Estoy cansada y veo que tu también estas cansado.
Noruego	Jeg er sliten og jeg ser at du er sliten	
Francés	Je suis fatigué et je vois que tu es fatiguée.	Je suis fatiguée et je vois que tu es fatigué.

Tabla 3.4: Importancia del género dentro de los tres idiomas EMPATHIC

se puede resumir en cuatro pasos:

1. Dividir la información suministrada por el DM que incluye etiqueta, subetiqueta, metadatos y un conjunto de parejas atributo-valor que puede estar vacío (los atributos obligatorios se incluyen en la metadata).

2. Transformar los valores de los posibles atributos de reemplazo predefinidos que aparezcan.

3. Seleccionar la plantilla adecuado en base a la información proporcionada por el DM. Primero atendiendo al idioma, posteriormente a la pareja etiqueta-subetiqueta y finalmente haciendo uso del total de atributos (obligatorios y posibles no obligatorios).

4. Sustituir en los huecos de la plantilla los valores de los atributos de remplazo correspondientes.

3.5.2. GROWsetta: la implementación

GROWsetta ha sido desarrollado en Python y está construido con la jerarquía de módulos que puede verse en la figura 3.13. Se puede ver como hay tres versiones para cada uno de los idiomas pero la estructura interna de cada versión es la misma. De hecho, los nombres de los módulos son iguales a los de castellano pero añadiéndole los sufijos *_fr* y *_no* para el francés y el noruego, respectivamente. Aparte de las tres versiones, se tiene un módulo principal que conecta con el DM para recibir la información, hace uso de una de las tres versiones para generar el texto y llama al TTS para sintetizar el audio a partir del texto generado.

El módulo principal

Como ya se ha dicho, el módulo principal (caja verde), GROWsetta_main.py está diseñado con tres objetivos diferentes. El primero es suscribirse a una cola donde el DM envía los DAs. Una vez se recibe dicha entrada, lo primero que se hace es extraer el idioma para crear un objeto de la clase GROWsetta a través de una llamada al módulo GROWsetta (caja azul) de la versión que corresponda. Seguidamente, se pone en marcha el proceso GROWsetta extrayendo las diferentes informaciones que el DM envía (etiqueta,subetiqueta, metadatos y slots) y se pasa a construir la frase. Todas estas funcionalidades son métodos de la clase GROWsetta definidos en el módulo que se va a explicar a continuación. Una vez generada la frase, ésta se incluye en la salida del NLG junto a los metadatos y se envía al TTS haciendo una llamado a su servicio. Tanto el primero como el tercer objetivo de dicho módulo vienen derivados de que el generador que se está implementando en este trabajo se va a integrar dentro del SDS del proyecto EMPATHIC y necesita estar en conexión con los módulos anterior y posterior.

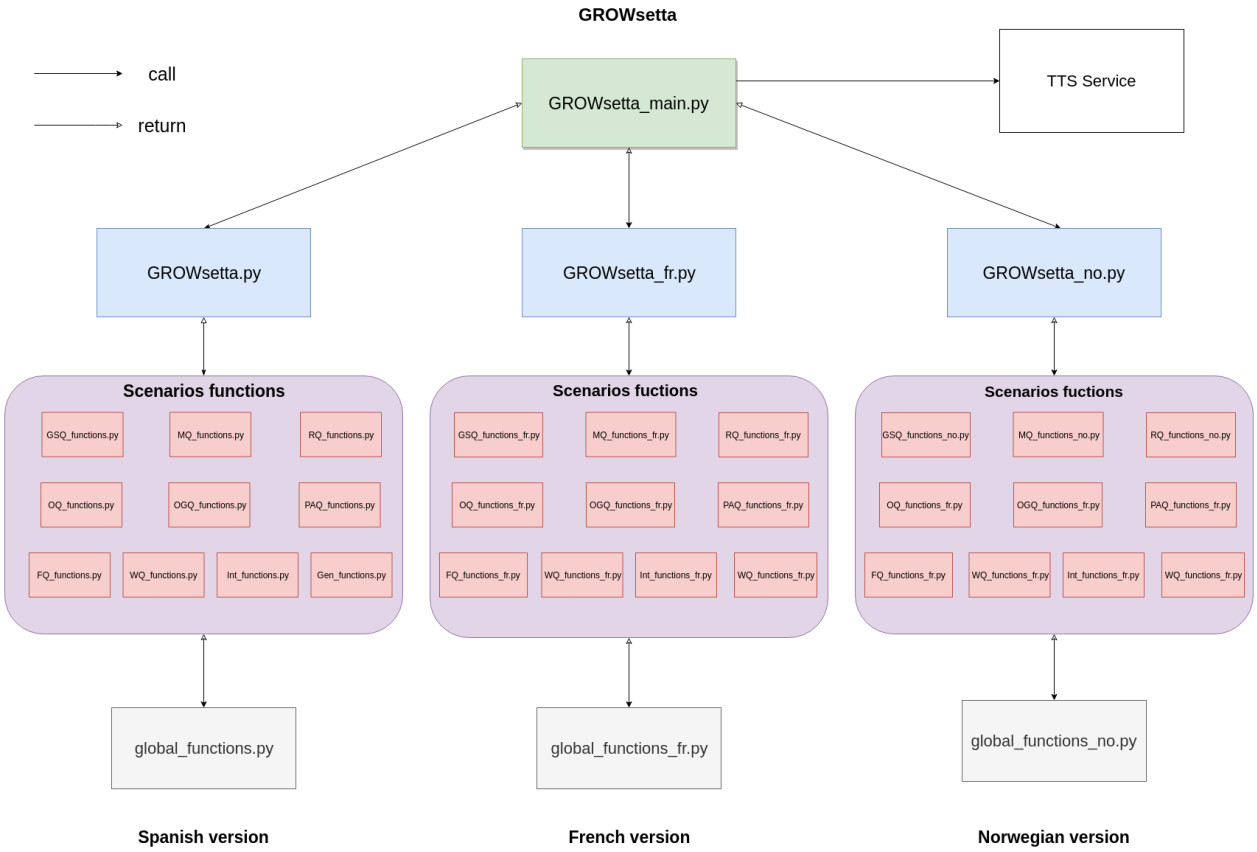

Figura 3.13: Jerarquía de módulos de GROWsetta

módulos GROWsetta

A partir de este momento en la explicación de la jerarquía de módulos entramos dentro de las tres versiones, lo que implica que la explicación de una de ellas sirve equivalentemente para las otras dos. Dicho módulo (caja azul en la figura) define la clase GROWsetta, en la que queda reflejada la estructura conceptual de GROWsetta mostrada en la figura 3.11.

El objeto GROWsetta tiene un elemento principal llamado árbol que se puede ver en la figura 3.14. A cada una de las ramas de ese árbol se accede con una de las las diez etiquetas principales de GROWsetta. Lo que se encuentra en cada una de las ramas es otra estructura en forma de árbol a la que se le ha denominado subárbol. A las ramas de dichos subárboles se accede a través las subetiquetas existentes para cada etiqueta principal, como se puede ver en el subtree Gen de la figura 3.14 . En las ramas de dichos subárboles se accede a las *scenario functions*. El número de dichas funciones en el total de GROWsetta es igual al número de posibles combinaciones que hay de etiqueta-subetiqueta, ya que para cada combinación de ese tipo se ha definido una función y éstas son las funciones de generación de frases como se verá después.

Fuera de dichas estructuras en forma de árbol, la clase GROWsetta presenta cuatro campos extras y tres funciones internas. Los cuatro campos extras son para guardar los datos que llegan del DM: etiqueta,subetiqueta, metadatos y slots. Por su parte, las tres funciones son aquellas que se usan en el módulo principal. La primera solo extrae la información y la introduce en sus respectivos campos. La segunda sirve para generar la frase y lo único que hace es llamar a una de las funciones escenario haciendo uso de las etiquetas y subetiquetas y pasándole como información el conjunto de slots. Finalmente, se tiene un método para reinicializar el objeto tras haber genera-

```
self.subtree_Gen={
"Echo":Gen_Echo,
"Agreement":Gen_Agreement,
"Disagreement":Gen_Disagreement,
"Doubt":Gen_Doubt,
"Pos_opinion":Gen_Pos_opinion,
"Neg_opinion":Gen_Neg_opinion,
"Other_opinion":Gen_Other_opinion,
"Apology":Gen_Apology,
"Apology_response":Gen_Apology_response,
"Thanks":Gen_Thanks,
"You_are_welcome":Gen_You_are_welcome,
"Pos_feedback":Gen_Pos_feedback,
"Neg_feedback":Gen_Neg_feedback,
"Congratulations":Gen_Congratulations,
"Condolance":Gen_Condolance,
"Hello":Gen_Hello,
"Goodbye":Gen_Goodbye,
"Advice":Gen_Advice,
"Inform":Gen_Inform,
"Open_question":Gen_Open_question,
"Yes_no_quest":Gen_Yes_no_quest,
"Explain_more":Gen_Explain_more,
"Open_topic":Gen_Open_topic,
"Close_topic":Gen_Close_topic,
"Select_topic":Gen_Select_topic,
"Topic_comment":Gen_Topic_comment,
}


self.tree={
"GSQ":self.subtree_GSQ,
"MQ":self.subtree_MQ,
"RQ":self.subtree_RQ,
"OQ":self.subtree_OQ,
"OGQ":self.subtree_OGQ,
"PAQ":self.subtree_PAQ,
"FQ":self.subtree_FQ,
"WQ":self.subtree_WQ,
"Gen":self.subtree_Gen,
"Int":self.subtree_Int,}
```

Figura 3.14: Estructura de árbol de la clase GROWsetta con el subárbol de Gen

```
def Gen_Hello(d_slots):
    d_slots=change_attributes(d_slots) #Change the values of some attributes
    templates=["¡Hola, hola!","Hola.",]
    if "<user_name>" in d_slots:
        templates+=2*["¡Hola <user_name>!","Bienvenido, <user_name>."]
    if "again" in d_slots:
        templates+=2*["Hola de nuevo."]
        if "<user_name>" in d_slots:
            templates+=3*["¡Hola de nuevo, <user_name>!"]
    template_select=random.choice(templates)
    sentence=fill_gaps(template_select,d_slots) #Filling the gaps with the values of the attributes.
    return (sentence)
```

Figura 3.15: Ejemplo de una función escenario

do la frase.

módulos de funciones escenario

Como se ve en la figura 3.13, las funciones escenario se reparten en 10 módulos diferentes para cada versión. Cada módulo está relacionado con cada uno de los 10 bloques de GROWsetta como se puede intuir a partir de los nombres de los módulos. Las funciones presentes en dichos módulos se utilizan para generar la frase haciendo uso del conjunto de slots atributo-valor. El proceso siempre es el mismo (figura 3.15), se cambian los valores de los atributos predefinidos haciendo uso de la función *change_attributes* de *global functions*, posteriormente se elige la plantilla atendiendo a la aparición de ciertos atributos. Una vez seleccionado la plantilla, se rellenan los huecos haciendo uso de otra de las funciones globales: *fill_gaps*. Con todo esto ya se tiene la frase construida para devolverla a los módulos superiores en la jerarquía.

```
def feel_alone(attribute,d_slots):
    attribute=attribute[:-1] #to delete the last ">"
    d_slots[attribute+">"]=random.choice(["no sentirse solo","sentirse acompañado","tener sensación de compañía"])
    d_slots[attribute+"_inv>"]=random.choice(["sentirse solo","no sentirse acompañado","no tener sensación de compañía"])
    d_slots[attribute+"_inv_pr_si_2_s>"]=random.choice(["te sientes solo","no te sientes acompañado","no tienes sensación de compañía"])
    d_slots[attribute+"_pr_si_2_s>"]=random.choice(["no te sientes solo","te sientes acompañado","tienes sensación de compañía"])
```

Figura 3.16: Ejemplo de función modificadora de los atributos predefinidos.

Charlie Chan	*is a*	*Chinese*	*restaurant*	*near*	*Cineworld*	*in the*	*centre of town*
Charlie Chan name inform	`inform`	**Chinese food inform**	**restaurant type inform**	`near inform`	**Cineworld near inform**	`area inform`	**centre area inform**
$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$	$t = 6$	$t = 7$	$t = 8$

Figura 3.17: Alineación de frase con el siguiente DA [40]
inform(name(Charlie Chan)type(restaurant)food(Chinese)near(Cineworld))

módulos de funciones globales

Por último, se pasa a explicar los módulos de funciones globales, que son uno por versión y tienen como finalidad definir funciones que puedan ser utilizadas indistintamente en cualquiera de los escenarios y que permiten la construcción de las frases. Contiene dos funciones: una previa a la selección de la plantilla y otra posterior. La previa a la selección de la plantilla modifica el valor independiente del idioma por el texto correspondiente incluyendo en el diccionario de slots atributo-valor todas las variantes definidas. Para cada valor predefinido de cada atributo tenemos una función modificadora de los valores a sus diferentes variantes (ver ejemplo en figura 3.16). Por su parte, la función posterior realiza la sustitución de los valores de los atributos en los huecos correspondientes a partir de la plantilla seleccionada y el conjunto de atributos.

3.6. TGen

A medida que el proyecto ha ido avanzando, la cantidad de datos de la que se disponía ha ido aumentando. Esto, unido a la predisposición a buscar un NLG que trabajase desde otro enfoque, ha hecho que se pasase a buscar un generador estadístico que se adaptase a las necesidades del proyecto. Dicha predisposición surge también de la necesidad de un sistema más fácilmente adaptable a varios idiomas y con una menor carga de trabajo a la hora de incluir nuevas frases o dominios. En esta ocasión, se ha decidido no realizar otro diseño desde cero y se ha hecho uso de una herramienta encontrada dentro de la bibliografía como es TGen*.

TGen es un generador estadístico desarrollado por Dusek y Jurcicek adaptado a un uso sobre SDS [14, 67, 66]. Aunque presenta tres distintas variantes, en todas ellas trabaja como un sistema entrenable a partir de representaciones semánticas no alineadas y sus correspondientes frases de salida. El hecho de que permita trabajar con entradas no alineadas permite eliminar el paso en el que se especifica que parte de la frase de salida está siendo utilizada para informar de un dato dado en el DA de entrada (figura 3.17), paso inevitable en otros generadores sin dicha capacidad [40, 68]. Otra de las ventajas que encontramos dentro de dicho sistema es que es adaptable a varios idiomas, ya que su funcionalidad principal es en inglés pero también se ha utilizado sobre algún conjunto de datos en checo**.

*https://github.com/UFAL-DSG/tgen
**https://github.com/UFAL-DSG/cs_restaurant_dataset

Analizando las bases de datos sobre las que TGen ha sido utilizado, también se vio que el tipo de DA utilizado en EMPATHIC era de una estructura similar o prácticamente idéntica a la que se usa en ellas. El esquema que siguen los DA se denomina *UFAL Dialog Act Scheme (UDAS)* [*] y sigue una metodología de construcción muy sencilla. La representación semántica de dichos DAs sigue una estructura compuesta por dos tipos de elementos: el tipo de acto de diálogo ,que sería un equivalente a nuestra pareja etiqueta-subetiqueta, y los slots valor-atributo. Partiendo de estos dos tipos de elementos, el DA para TGen se construye de la siguiente manera: el DA es una secuencia de elementos que se denominan DA ítem (DAI). El número de ítems de cada DA es igual al número de atributos del DA salvo que este sea 0. Si es 0, el DA tendrá un formato con un único DAI sin atributos: *DA_type()*. Para el resto de casos, el formato de cada DAI sería *DA_type(attribute=value)* unido al resto de DAI a traves del caracter *&* . Así el DA de la figura 3.17 quedaría de la siguiente manera:

inform(name=Charlie Chan)&inform(type=restaurant)
&inform(food=Chinese)&inform(near=Cineworld)

Dicha estructura permite una adaptación sencilla del formato EMPATHIC de DA al necesario para trabajar con la herramienta TGen. A continuación, se va a pasar a explicar las tres versiones de TGen que Dusek y Jurcicek han implementado.

3.6.1. TGen basado en búsqueda A*

La primera versión del generador de TGen trabaja de forma tradicional, ya que lo hace en dos fases (figura 3.18): planificación de sentencias y realización [14]. En la primera fase se transforman los DA en arboles sintácticos que forman el plan de sentencia. Esta transformación está basada en el algoritmo A*. Mientras que para el paso a lenguaje natural se hace uso de un realizador basado en reglas. Por lo que la parte estadística del generador aparece en la primera fase.

El planificador de sentencias está basado en el algoritmo A*. Dicho algoritmo fue desarrollado por Hart et al. en 1968 [69] como un algoritmo de búsqueda en grafos de tipo heurístico que tenía como objetivo la búsqueda del camino de menor coste entre un nodo y otro del grafo. En esta ocasión, la idea se asemeja a la del algoritmo original, ya que lo que se hace básicamente es ir añadiendo nodos al árbol sintáctico hasta encontrar el árbol óptimo.

Durante el proceso se trabaja con dos componentes: un generador de candidatos que crea posibles arboles añadiendo iterativamente nodos y un evaluador que asigna un valor a lo adecuado que es el árbol generado. Y, al igual que sucede con el algoritmo original, se conservan dos tipos de conjuntos, que denominamos abierto y cerrado.

El proceso comienza con el conjunto abierto con un árbol vacío (solo el nodo origen del árbol) y el conjunto cerrado vacío. Tras este punto de partida comienza un ciclo de cuatro pasos:

1. Se selecciona el mejor candidato del conjunto abierto y se añade al conjunto cerrado (en el caso inicial se escoge el árbol vacío).

2. Se genera un conjunto de candidatos añadiendo un nodo al árbol escogido en el proceso anterior (figura 3.19.

3. Se puntúan todos estos posibles candidatos y se añaden al conjunto abierto siempre que no formen parte ya del conjunto cerrado.

4. Se analiza si alguno de los sucesores mejora el mejor candidato del conjunto cerrado.

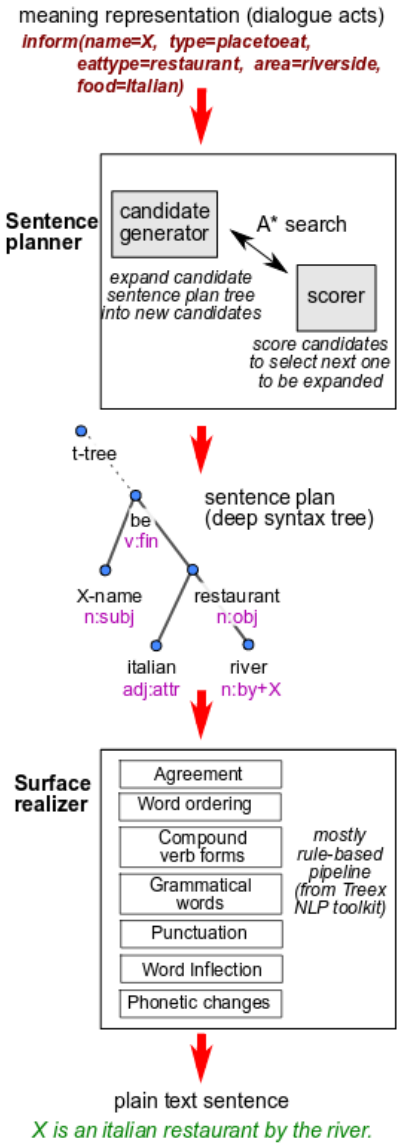

Figura 3.18: Estructura general del generador TGen basado en A* [14]

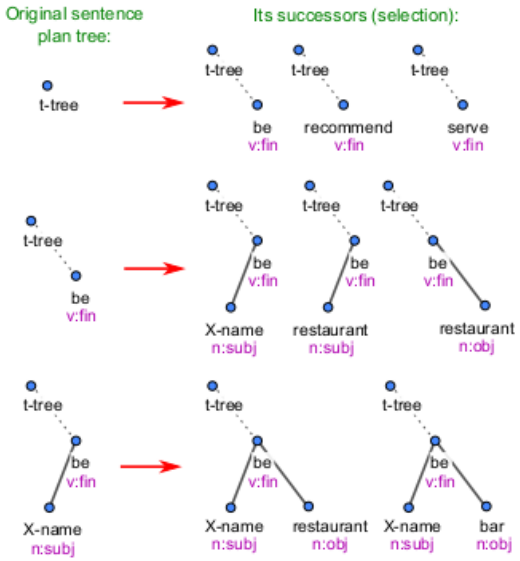

Figura 3.19: Ejemplo del proceso de generación de candidatos [14]

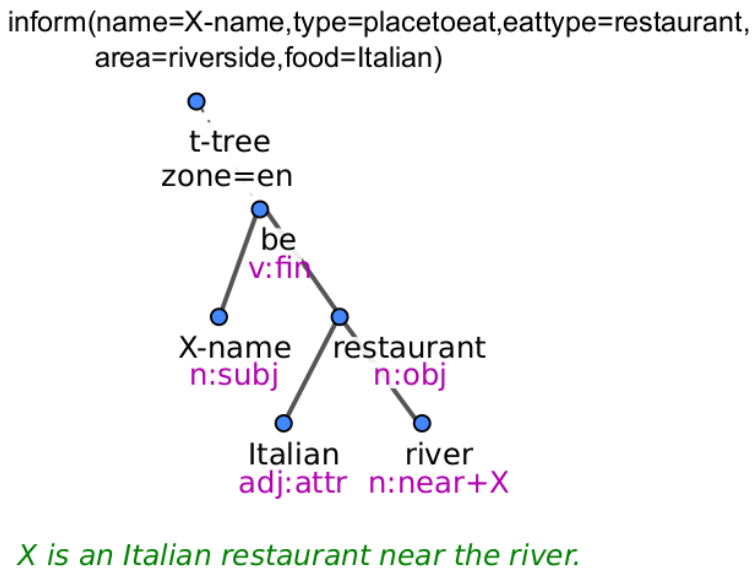

Figura 3.20: Ejemplo de los dos tipos de salida de TGen [67]

Dichos cuatros pasos se repiten hasta que se da la condición de parada. La condición de parada para este algoritmo tiene dos versiones: establecer un número máximo de veces consecutivas en los que los sucesores no mejoren el candidato óptimo del conjunto cerrado o cuando el conjunto de candidatos esté vacío. El árbol escogido es el de mayor puntuación de ambos conjuntos. Para su posterior transformación a lenguaje natural, Dusek y Jurcicek utilizan un sistema basado en reglas pero cualquier realizador que tenga la capacidad para partir de arboles sintácticos profundos podría servir.

3.6.2. TGen basado en seq2seq

En esta versión de TGen se plantea un generador que da lugar a dos posibilidades distintas: la de generar arboles sintácticos o la de generar frases (figura 3.20) [67]. El generador trabaja en ambos casos bajo un enfoque seq2seq pero con la capacidad de trabajar como planificador de sentencias o como un generador E2E donde ambas fases se realizan en un solo paso. Dicho generador con enfoque seq2seq se basa en una arquitectura RNN con un encoder-decoder que opera con secuencias de longitud variable.

Por las características de los datos utilizados y las necesidades del proyecto, la versión de TGen que se ha utilizado es la característica E2E del generador, ya que lo que interesa en este caso es generar frases directamente.

Tokenización de las entradas y salidas de la red

A la hora de presentar las distintas entradas (DAs) y las salidas (frases o árboles), se va a hacer a través de una secuencia de tokens, lo que viene derivado de su arquitectura seq2seq. Cada token a su vez presenta una representación en formato embedding. Dicha representación es una representación vectorial en un espacio continuo con una serie de ventajas frente a las representaciones clásicas en un espacio discreto, destacando por encima de todas el buen sentido geométrico que hace que elementos similares tengan una representación similar [70].

*https://github.com/UFAL-DSG/alex/blob/master/alex/doc/ufal-dialogue-acts.rst

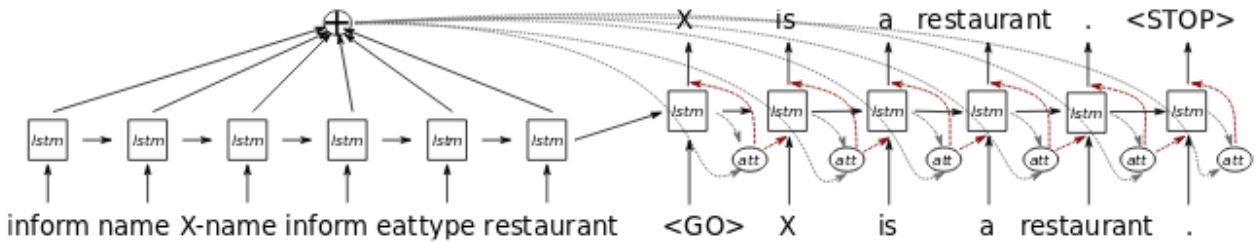

Figura 3.21: Arquietectura de la red de TGen [67]

La tokenización de los DAs viene a darle sentido al formato de los DAs para TGen, ya que al igual que el formato de los DAI es el formato también de la secuencia de tokens de la red. Cada secuencia de tokens de entrada está formado por un triplete (tipo de DA, atributo, valor) para cada slot presente en el DA. Sin embargo, cada triplete no se considera un token en sí, sino que cada elemento del triplete es un token independiente con una representación embedding distinta. Por su parte, la tokenización a nivel de frases es la habitual ya que se considera cada palabra o signo de puntuación como un token.

Arquitectura de la red

La arquitectura de la red (figura 3.21) utilizada en TGen es una red seq2seq con atención. Dicha red trabaja con dos RNNs, una como encoder y otra como decoder. Dicha idea está tomada teniendo como referencia ciertos trabajos donde dicho tipo de red se utilizaba como un sistema de traducción automática estadístico [71, 72, 73]. El hecho de que dicha red sea adaptable a estas dos aplicaciones no es raro porque ya hemos comentado que el NLG se puede entender como un tipo de traducción, en la cual se trabaja desde el idioma de DAs a lenguaje natural. Una de las principales ventajas de esta arquitectura es que tiene la capacidad de traducir o generar lenguaje natural a la vez que realiza el alineamiento.

Una RNN estándar se puede definir como una red que, a partir de una secuencia de vectores de entrada $\mathbf{x} = (\vec{x}_1, ..., \vec{x}_T)$*, es capaz de generar una secuencia de salida $\mathbf{y} = (\vec{y}_1, ..., \vec{y}_{T'})$**. En este proceso, a cada paso t la red actualiza un estados oculto $\vec{h}_t = f(\vec{h_{t-1}}, \vec{x}_t)$, donde f es una función no lineal que en este caso va a ser una celda Long Short Term Memory (LSTM) o una Gated Recurrent Unit (GRU). Dichas redes RNN estándar tienen la capacidad de operar de manera correcta si el alineamiento entre las redes estaba previamente definido pero los problemas surgían al trabajar con secuencias de longitud variable o con alineamientos no definidos previamente.

Para abordar este problema, Cho et al. propuso la idea de hacer uso de un encoder y un decoder, ambos basados en RNNs (figura 3.22). De este modo, se codifica la entrada en un vector de representación de longitud fija para posteriormente a partir de dicho vector generar la secuencia de salida.

Así, el codificador es una RNN que lee la entrada secuencialmente y actualiza los estados ocultos de la manera habitual. A partir de dichos estados ocultos, se construyen los T' vectores de contexto $\vec{c}_t = \sum_{i=1}^{T} \alpha_{t_i} h_i$, que dan lugar al mecanismo de atención que captura el alineamiento y que posteriormente se usan en la parte del decodificador. Dicho alineamiento viene representado con los α_{t_i} que establecen cuanto aporta el elemento i-ésimo de la secuencia de entrada al elemento t-ésimo en la salida.

*En este trabajo, cada $\vec{x}_t$ es el embedding asociado a cada uno de los elementos del triplete antes presentado

**$\vec{y}_t$ es el embedding de cada palabra o signo de puntuación presente en la frase de salida, incluyendo también embeddings de los símbolos especiales de inicio y final de secuencia.

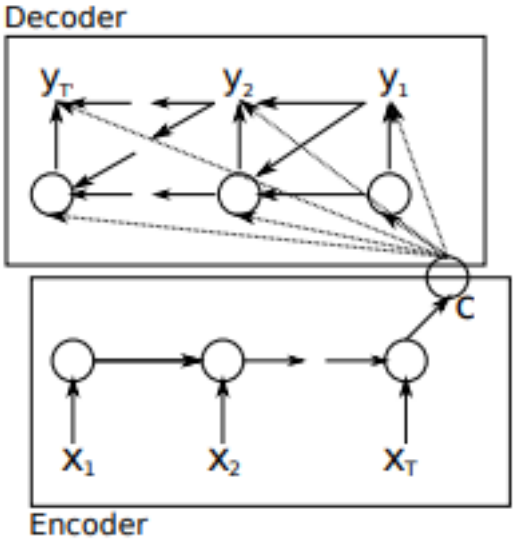

Figura 3.22: RNN encoder-decoder [72]

Por su parte, el decodificador es otra RNN con la función de predecir el siguiente elemento de una secuencia $\vec{y}_t$ teniendo en cuenta la secuencia predicha hasta entonces y la secuencia de entrada. La probabilidad de que la palabra a predecir sea una concreta se define como:

$$P(\vec{y}_t|\vec{y_{t-1}}, \vec{y_{t-2}}, ..., \vec{y}_1, \mathbf{x}) = softmax((\vec{s}_t \circ \vec{c}_t)W_Y)$$

donde s_t son los estados ocultos del decodificador. Dichos estados se calculan de forma diferente a la que se ha explicado en las RNN estándar. Como se puede ver en la figura 3.22, el cálculo de los estados ocultos (círculos blancos en la figura) implica al vector de contexto en el decodificador:

$$\vec{s}_t = f((\vec{y_{t-1}} \circ \vec{c}_t)W_S)$$

salvo para el estado oculto inicial del decodificador, donde se recoge como valor el último estado oculto del codificador ($s_0 = h_T$).*

Con todo esto, en el entrenamiento de la red se trata de maximizar de forma conjunta entre ambas fases el siguiente log-likelihood:

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} log\, p_\theta(y_n|x_n)$$

donde θ es el conjunto de parámetros que hay que estimar y (x_n, y_n) es cada una de las N parejas entrada-salida de entrenamiento. Como todo es diferenciable en dicho proceso, para la estimación de los modelos TGen permite la utilización de distintos modelos de optimización como el habitual gradient descent, AdaGrad [74] y Adam [75].

Otra de las características de dicha arquitectura seq2seq es que a la hora de la búsqueda de la secuencia de salida hay dos modalidades: greedy decoding y beam search. Sin embargo, greedy decoding es en realidad un caso concreto de lo que se entiende como beam search. En la búsqueda beam search, se expande la secuencia de izquierda a derecha guardando aquellas k secuencias con mayor probabilidad, expandiendo a cada paso aquellas en las que no haya aparecido el símbolo de final de secuencia. De este modo, a cada paso tenemos k posibles secuencias a expandir (salvo

*En las dos ecuaciones anteriores $\circ$ es el operador de concatenación de vectores, mientras que W_Y y W_S son matrices de proyección.

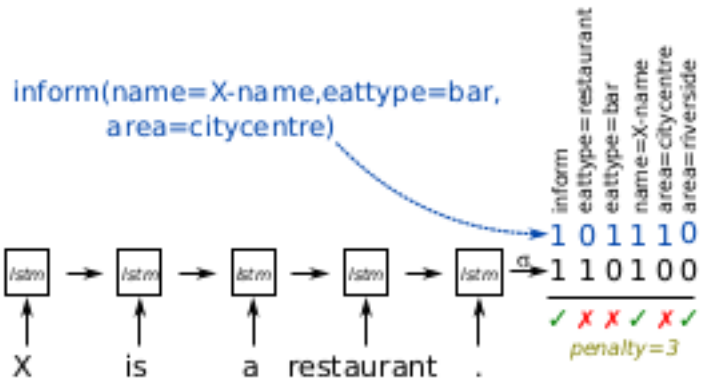

Figura 3.23: Reranker de TGen [67]

las ya completas), dando lugar a k^2 nuevas secuencias ya que de cada secuencia se expande con k diferentes opciones y de esas para el siguiente paso se conservan las k mejores. En el caso del greedy decoding, a cada paso expandimos una única secuencia con la palabra más probable. Esto quiere decir que greedy decoding es básicamente un beam search con k=1.

El reranker

La herramienta TGen permite la incorporación de un reranker (figura 3.23) que se trata de un clasificador que tiene como objetivo reevaluar las k salidas del beam search. La idea es asegurar que la salida presenta todos y únicamente los elementos semánticos presentes en el DA de entrada, para lo que se penaliza la no presencia de dichos elementos y la aparición de otros no presentes en el DA.

Para calcular dicha penalización convertimos tanto el DA de entrada como las sentencias generadas en vectores binarios donde se representa si los diferentes tipos de actos de diálogo y los diferentes slots encontrados durante la fase de entrenamiento se encuentran dentro del DA y frases sobre las que se trabaja en ese momento. Una vez obtenida dicha binarización, la penalización se calcula como la distancia Hamming entre dichos vectores. En la figura 3.23 se puede ver el proceso de binarización de entrada y salida y el cálculo de la penalización. La penalización propia de cada frase se sustrae de la probabilidad asociada que tenía .

En dicho proceso debemos transformar tanto DAs como frases en vectores binarios. En el caso de los DAs, la transformación es directa, ya que solo hay que analizar si los tokens asociados a cada tipo de DA, atributo o valor aparecen en la secuencia de entrada. Por su parte, la binarización de las frases se hace a través de una red del mismo tipo que el encoder de la red principal, es decir, una red RNN estándar con celdas LSTM, con la diferencia de que al último estado oculto se le aplica una celda sigmoide para realizar clasificación binaria que determine si dicha frase tiene la presencia de un determinado DA type o slot. Para el entrenamiento de esta red, las entradas y salidas son las inversas a las de la red principal, de tal modo que para cada frase (entrada en esta red) se aprende qué DA (salida en esta red) tendría que tener asociado. Así, posteriormente la red trabaja generando un DA para la frase dada y realmente la distancia Hamming se obtiene analizando las diferencias entre el DA real y el obtenido con la red del reranker.

3.6.3. TGen basado en seq2seq con contexto

La última versión de TGen diseñada por Dusek y Jurcicek [66] es una adaptación de la versión anterior de la arquitectura seq2seq para trabajar con contexto. Se trata de una versión muy in-

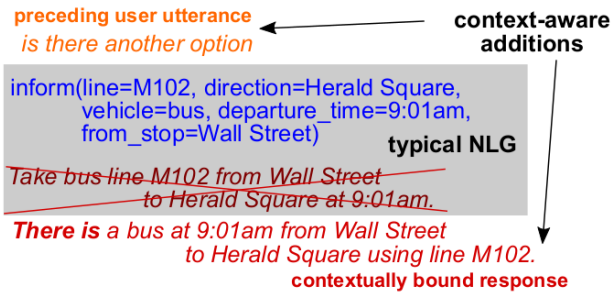

Figura 3.24: Ejemplo que muestra la diferencia de la generación de TGen con o sin contexto [66]

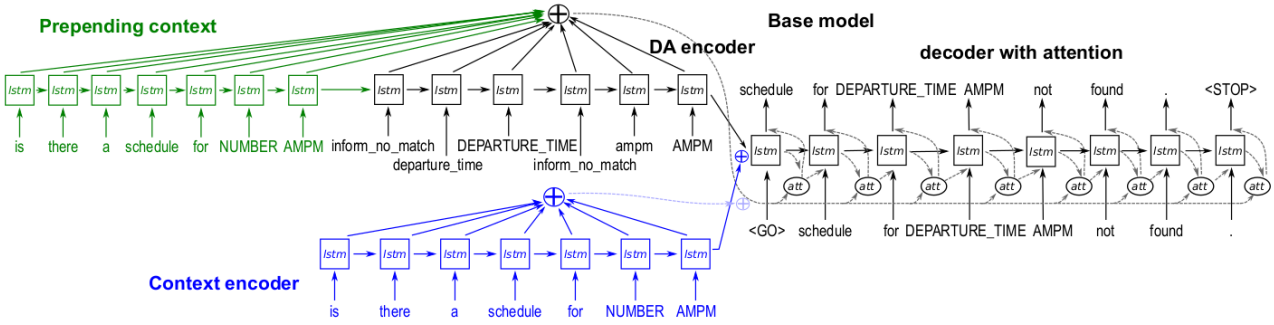

Figura 3.25: Arquitectura de la red para el uso de TGen con contexto [66]

teresante a la hora de integrarla en un SDS, ya que permite una mayor adaptabilidad al usuario al tener en cuenta la última frase dicha por éste al generar la respuesta (figura 3.24). Se trata de una forma de añadir naturalidad a la respuesta, ya que lo habitual dentro de una conversación es que la forma de expresarse de cada uno de los participantes esté en parte condicionada por la forma de hablar del otro. La razón de no haber utilizado esta configuración en este trabajo es que los datos no estaban adaptados a este posible uso. Sin embargo, la idea es incorporar en un trabajo futuro el uso de contexto a la hora de aplicar TGen.

Como se puede ver en la figura 3.25 la base de la red utilizado en este caso es la arquitectura en la versión anterior de TGen (lo que aparece en negro en la figura). De hecho, solo se han modificado tres elementos de una versión a la otra. Dos de ellos son los que aparecen marcados en azul y verde en la figura, por lo que están relacionados con la arquitectura de la red. Por su parte, el último cambio tiene que ver con el reranker.

El primer cambio dentro de la arquitectura de la red aparece en la secuencia con la que va a trabajar el encoder de la red. Como vemos en verde en la figura lo que se hace es añadir de forma previa la frase de contexto al DA como secuencia de entrada del decoder. A la hora de operar, contexto y DAs trabajan con una tokenización separada.

Otro elemento que se añade es un codificador únicamente del contexto. El codificador es del mismo tipo del ya explicado. Posteriormente los estados de ambos codificadores se concatenan para su uso en el decodificador, por lo que el decodificador trabaja con vectores de doble tamaño.

Finalmente, se añade un segundo reranker con la idea de analizar si aparecen palabras utilizadas en el contexto dentro de la salida. A diferencia del otro reranker, en vez de penalizar las frases lo que se hace en este caso es aumentar la probabilidad de las frases que presentan elementos del contexto en su formación.

Capítulo 4

Resultados

En este capítulo se van a detallar los resultados de los distintos procesos seguidos en la construcción del EMPATHIC-NLG. A grandes rasgos son dos los procesos seguidos: la preparación de los datos y el diseño de los generadores. A nivel de datos, se van a presentar los datos estadísticos referentes a la anotación de la base de datos. Por su parte, en los generadores se analiza la calidad de su salida en base a distintos criterios.

4.1. La base de datos

En esta sección se presentan las estadísticas del conjunto de datos con el que se ha trabajado. Se inicia presentando la base de datos en castellano y posteriormente se centra el análisis en los otros dos idiomas: francés y noruego. Como se menciona en la sección 3.4 a la hora de realizar la anotación de los datos en castellano, la base de datos se dividió en dos conjuntos: el conjunto de anotación propia y el conjunto que incluye las conversaciones con usuarios reales anotado por los anotadores contratados. Por ello, se ha analizado tanto el total de los datos como ambos por separado para ver sus diferencias a nivel estadístico. En cuanto a los otros idiomas, solo se han realizado anotaciones de las conversaciones reales siguiendo el sistema de etiquetado llevado a cabo por los anotadores. Por ello, en la comparativa solo se han utilizados los datos equivalentes en castellano, es decir, los etiquetados por los anotadores.

4.1.1. Datos en castellano

El conjunto de datos en castellano es el más amplio de todos, ya que no solo el número de usuarios reales que han tenido una conversación con el VC es mayor sino que además contamos con otro conjunto de datos que se han obtenido de fuentes diferentes (conversaciones inventadas por el coach, transcripción de una sesión real de una persona con el coach profesional,...). Con todo esto, el número total de datos en castellano es de 8173 (tabla 4.1).

El conjunto de anotación propia fue construido tomando información desde diferentes fuentes (word, pdf, transcripciones,...) en las que se daban posibles frases que podían estar presentes en una sesión. Al realizar el procesado de todos estos datos se hizo de tal manera que lo que se consideraba como unidad a analizar fuese ya lo que previamente se ha definido como sentencia,

Total	**Anotación propia**	**Anotación anotadores**
8173	2188	5985

Tabla 4.1: Número de sentencias para el castellano.

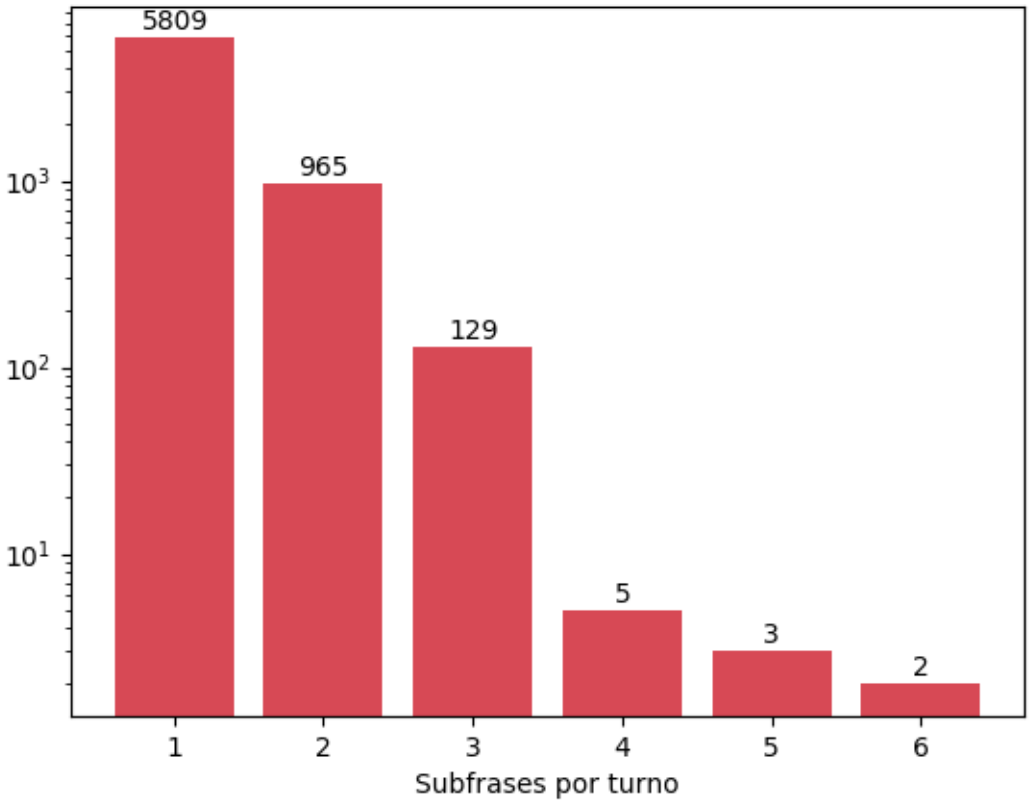

Figura 4.1: Número de subfrases por turno en las conversaciones reales en castellano

es decir, la menor partícula del texto que proporcione una información. Por lo que uno de los análisis que no se hace con este conjunto es analizar el número de sentencias por turno (que sí que se hace con las conversaciones reales). Este conjunto está formado por 2188 sentencias.

Por su parte, la anotación llevada a cabo por los anotadores fue realizada sobre conversaciones reales de usuarios con el VC, el cual operaba con un sistema de WOZ en el que una persona (*Wizard*) respondía de forma remota haciendo pensar a los usuarios que el sistema trabajaba de forma automática. El total de conversaciones que finalmente se pudieron realizar de las 180 que se tenía como objetivo fueron 142. El número de turnos totales de usuario en esas 142 conversaciones es de 4725, lo que hace una media de 33 turnos de usuario por conversación. En dichos turnos, lo habitual es que el número de sentencias sea solo una pero no siempre se cumple. Así, el número total de sentencias en este dataset es de 5985, lo que deja una media de subfrases por turno no muy superior a uno. En la figura 4.1 se puede ver como la gran mayoría de turnos están compuestos por una única frase. De hecho, el número de turnos con más de una subfrase sumando todos los casos ocurridos (de 2 a 6) es en torno a un tercio de los que solo tienen una.

Distribución de las etiquetas

A nivel de análisis por el tipo de dato, la gran diferencia entre un conjunto y otro es que en uno se partía de turnos en los que se podía hacer una división de estos en subfrases mientras que el otro se partía de los propios sentencias a la hora de realizar la anotación. Por lo tanto, a partir de aquí toda estadística que se extraiga de un conjunto también se puede obtener del otro. De este modo, todas las características de la anotación que se han considerado se presentan para ambos

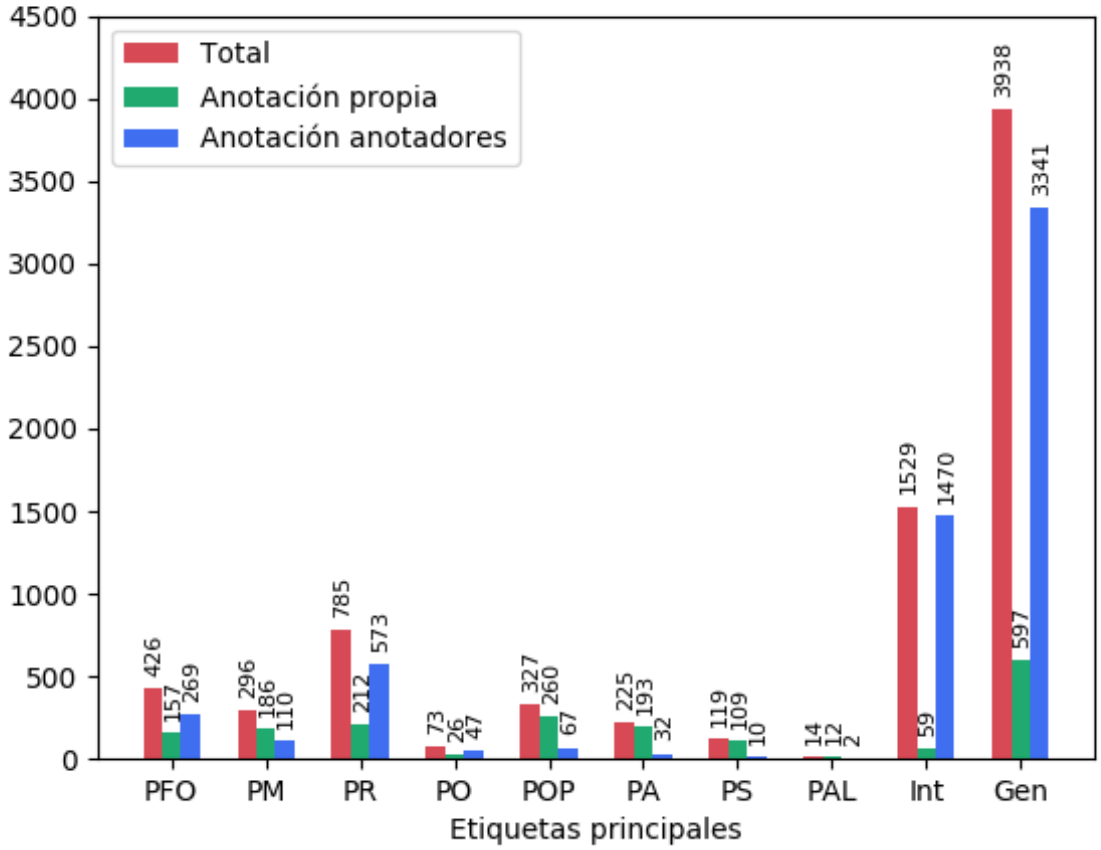

Figura 4.2: Distribución de las etiquetas principales para los datos en castellano.

conjuntos de datos y, por tanto, para el total de los datos.

Un primer paso para analizar la anotación es ver cómo se distribuyen las frases en función de las etiquetas principales. Como se puede ver en la figura 4.2, para el conjunto de datos total destaca el uso de frases dentro del escenario General, lo que no es nada extraño ya que son todas aquellas frases de carácter menos específico y que aparecen en cualquier tipo de conversación. Seguido, se encuentran las frases que se engloban dentro del escenario de Introducción. Este hecho se debe a que la estructura de las fases de inicio es muy marcada y es por ello que la gran mayoría de las frases realizadas en la fase de introducción se hayan etiquetado de esta manera. Fuera de estos dos escenarios, quedan las etiquetas del modelo GROW, donde las preguntas para testear la realidad (PR) han sido las más utilizadas, dejando en un segundo plano tanto las preguntas previas de motivación (PM) y de definición de un objetivo (PFO) como las posteriores de generación de opciones (POP) y de establecimiento de un plan (PA). De forma reducida se encuentran las preguntas de obstáculos (PO) y de seguimiento (PS) y de forma prácticamente nula las preguntas de alerta (PAL).

De la comparativa entre las distribuciones en los dos conjuntos datos y el total (figura 4.2) se aprecia como en ambos conjuntos por separado el caso general sigue siendo el predominante, pero hay algunas etiquetas que presentan unas diferencias interesantes con respecto al total. Comenzando con el escenario de introducción se puede ver que casi la totalidad de los DAs de esta etiqueta aparecen dentro de las conversaciones reales. Eso se debe a que en el primer dataset solo se contaba con una serie de ejemplos muy concretos para establecer lo que era una sesión de

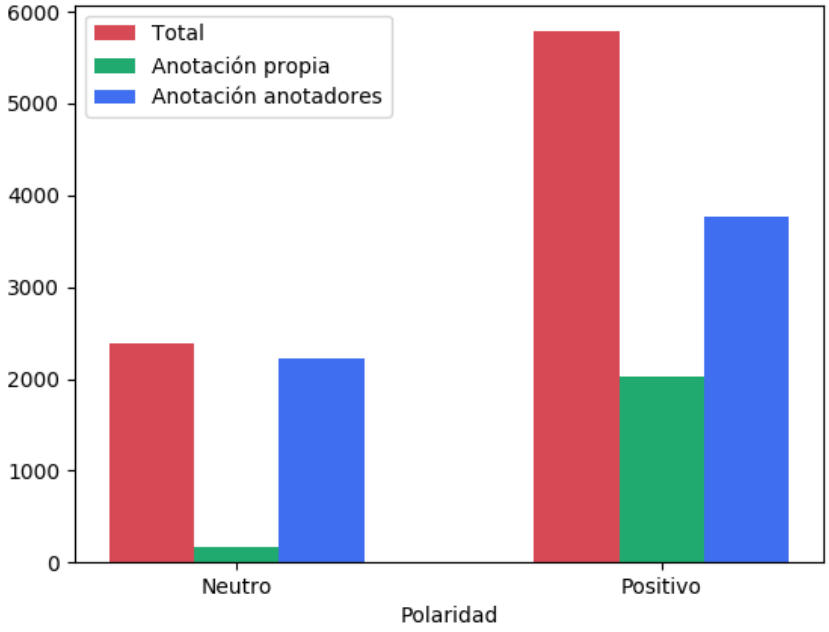

Figura 4.3: Distribución de la polaridad para los datasets en castellano.

inicio pero luego se ha podido ver como dicha pauta se ha seguido en las sesiones reales y por ello que la segunda etiqueta mayoritaria a nivel global sea esa. Dentro del modelo GROW, se puede ver como hay pequeños cambios afectados por las características de las sesiones reales. En dichos encuentros normalmente al *wizard* le ha costado superar las dos primeras fases en los 10 minutos de los que disponían y es por ello que las 3 etiquetas más utilizadas sean las referentes a esas dos primeras fases (PFO, PM y PR). En el resto de etiquetas del modelo GROW, la mayoría de frases con dichas etiquetas aparecen en el primer dataset, donde las conversaciones eran no reales (diálogos inventados) o realizadas por un coach profesional con la capacidad y tiempo para llevar la sesión más lejos.

Polaridad

Los resultados obtenidos en función de la polaridad muestran, como era de esperar, que la polaridad positiva (polaridad por defecto para el VC) aparece en más frases de las que en lo hace con polaridad neutra(figura 4.3). Dicha descompensación entre las dos polaridades queda más ampliamente reflejada en el primer dataset con un porcentaje mayor al 80% asociada a la polaridad positiva. El hecho de que en conversaciones reales también dicha polaridad se establezca por encima del 60% deja claro que sí debe ser la polaridad por defecto. De hecho, en el global de las anotaciones las frases con polaridad positiva son más del doble que las que aparecen con polaridad neutra.

Géneros de agente y usuario

Otro de los elementos que se pedía determinar durante el etiquetado era si las frases a etiquetar se podían realizar sin necesidad de atender al género de los participantes en la conversación o si, por lo contrario, el género de agente y/o usuario quedaba reflejado en la frase y solo se podía utilizar dicha frase en aquellos casos que los géneros concordasen con los extraídos de la frase. Lo observado es que tanto para el género del agente como del usuario, el 98% de las frases se pre-

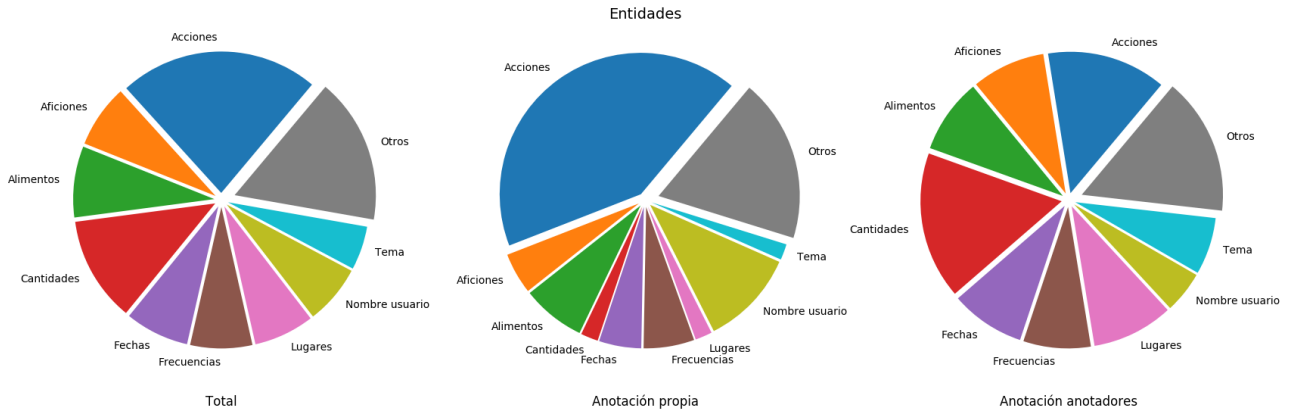

Figura 4.4: Comparativa de las entidades identificadas en los datasets en castellano.

sentan con género indiferente. Solo en 51 del total el género del coach es identificable y en 92 lo es para el género del usuario. De este modo, casi en su totalidad las frases de los NLG que se han implementado no van a atender a los géneros a la hora de generarse.

Entidades

Por último, la capacidad de incluir entidades identificadas en el NLU en las respuestas era una de las capacidades que se quería que presentase el NLG-EMPATHIC. A la hora de anotar se han identificado aquellos elementos que se podían extraer como entidades en el NLU y que se podrían haber incluido en la frase en base a que se hubiese suministrado como parte de los slots del DA. El análisis de las entidades que aparezcan no modifica la forma en la que se ha implementado el NLG o no sirve para tomar decisiones como la de cuál es la polaridad por defecto pero sí que sirve para ver qué tipo de entidades aparecen más, sobre todo con mayor interés en las que los anotadores han visto en las conversaciones reales.

En base a lo observado en la figura 4.4, cabe destacar la alta presencia de *Acciones en infinitivo*, sobre todo en el primer dataset, aunque en el segundo dataset también aparece de forma mayoritaria. Todas estas acciones son acciones que definen objetivos, opciones, realidades o obstáculos del usuario y por ello es interesante que aparezcan en las preguntas del coach, ya que eso asegura haber identificado alguno de esos elementos importantes en el módelo GROW. Todos aquellos modificadores de dicha acción (cantidades,cantidades de tiempo, frecuencias, fechas,lugares,..) son las siguientes entidades en número de apariciones, destacando por encima de todos *Cantidades* que es la entidad mayoritaria en las conversaciones reales. Que dicha entidad y también *Alimentos* sean tan relevantes en las conversaciones reales tienen su explicación en que la temática de las segundas sesiones con el VC han sido de nutrición.

4.1.2. Datos en noruego y francés

Los datos con los que se han contado para el noruego y el francés han sido recogidos en las sesiones de WOZ con usuarios reales. De este modo, todo lo que se analice se va a poder comparar entre los tres idiomas pero en el caso del castellano sólo se va tener en cuenta el segundo dataset para que todo sea más equiparable. Evidentemente, los anotadores utilizados para las tareas en estos idiomas son personas que tienen como lengua nativa dichos idiomas, al igual que para el castellano lo fueron personas nacidas en España.

Idioma	Número de conversaciones	Total turnos	Turnos por conversación	Frases totales
Castellano	142	4725	33	5985
Francés	68	2384	35	3027
Noruego	62	1320	21	2002

Tabla 4.2: Estadísticas de las conversaciones reales en los 3 idiomas.

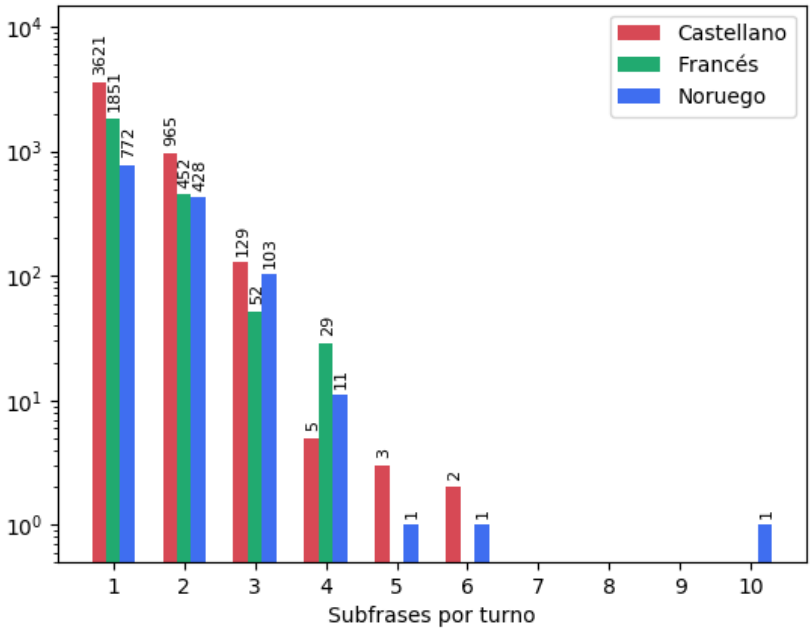

Figura 4.5: Comparativa de las subfrases por turnos en los tres idiomas.

Como se aprecia en la tabla 4.2, solo en el caso de Francia se ha superado las expectativa de sesiones a realizar (68 de las 60 previstas). En España y Noruega se tenían previsiones de 180 y 120, respectivamente, y no se ha alcanzado. Sin embargo, son datos suficientes para realizar un análisis de ellos. El primer dato destacable es cómo las conversaciones realizadas en noruego presentan menos intervenciones por parte de los coach, con una media de 21 turnos, lejos de los más de 30 de los otros dos idiomas. En cuanto al número de subfrases (figura 4.5) podemos ver cómo, al igual que sucedía en castellano, predominan los turnos no divisibles en más de una sentencia. Así, en ambos casos la media de subfrases etiquetadas por turno es ligeramente superior a uno.

Distribución de las etiquetas

Los resultados reflejados en la figura 4.6 dejan una serie de observaciones. Se pueden encontrar ciertas similitudes como que los escenarios General y de Introducción son los dominantes, siendo el primero de ellos siempre la etiqueta para más del 50 % de los datos. Aparte de la aparición de la etiqueta de no clasificable (N.C.) y de la no presencia de alguna de ellas en noruego (PAL y PS), la principal diferencia es la etiqueta que ocupa el tercer lugar, donde en el caso de Francia y España dicha etiqueta era aquella con la que se definía el estado actual del usuario (PR), mientras que en las sesiones hechas en Noruega lo que se observa es que las sesiones no han avanzado hasta dicha fase ya que la pregunta del modelo GROW que más se repite es PFO. El hecho de que las sesiones en noruego se queden en la fase inicial de fijar el objetivo viene en concordancia con el menor

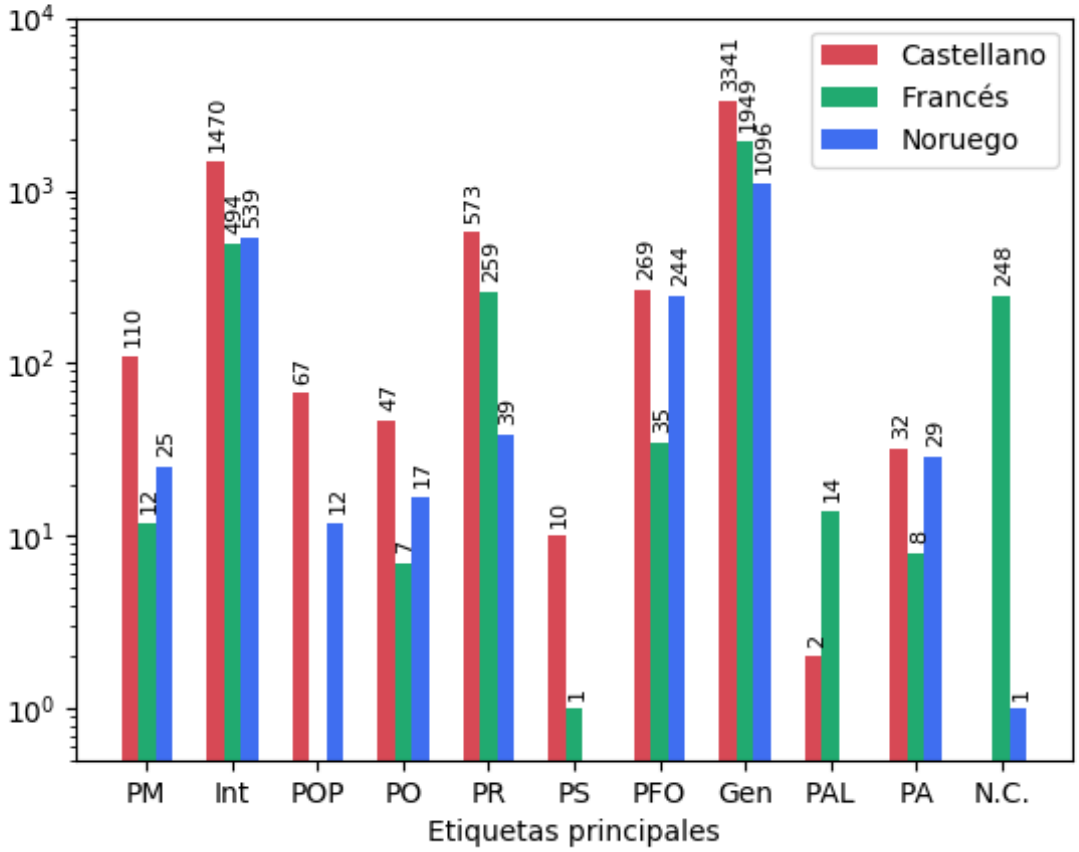

Figura 4.6: Comparativa de la distribución de etiquetas en los tres idiomas.

número de intervenciones del coach durante la sesión.

Polaridad

Una de las grandes diferencias que presentan las anotaciones hechas fuera de España con las realizadas por los anotadores españoles es que la polaridad mayoritaria es la neutra (figura 4.7). La realidad no es que los diálogos en dichos idiomas tengan una polaridad de carácter más bajo sino que al realizar las anotaciones se ha considerado como polaridad por defecto la neutra, en vez de la positiva. Por ello, si se quiere considerar la polaridad a la hora de implementar los generadores estos datos deben ser reetiquetados de algún modo.

Géneros de agente y usuario

En noruego este análisis no tiene sentido, ya que, al igual que sucede en idiomas como el inglés, no se ven afectadas las frases por los géneros de los interlocutores. En cuanto al francés, como sucede con el español, las frases en las que se puede determinar el género del agente y/o el usuario no supera el 2 %. De este modo, los generadores en los 3 idiomas van a partir de datos en los que prácticamente no va a aparecer la problemática de la concordancia de los géneros reales y los de la frase.

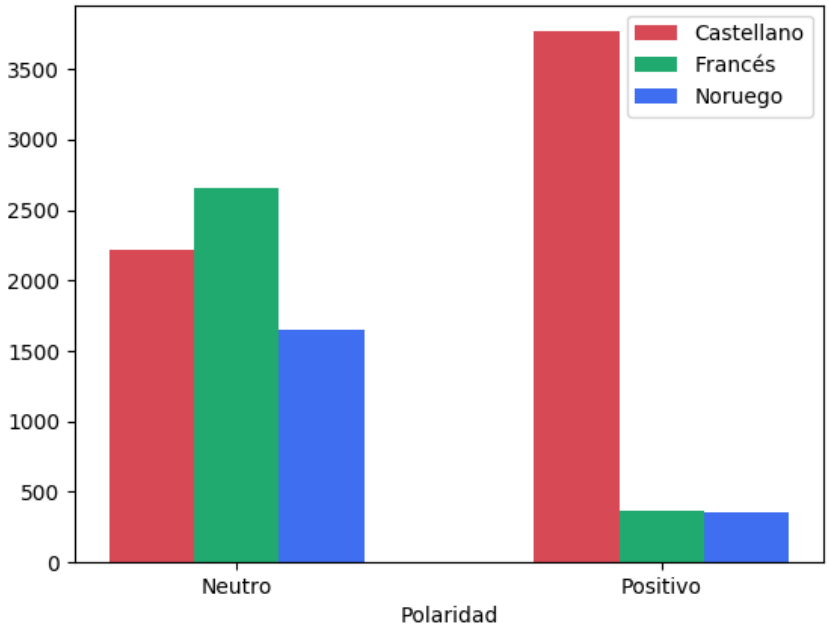

Figura 4.7: Comparativa de la distribución de la polaridad en los tres idiomas.

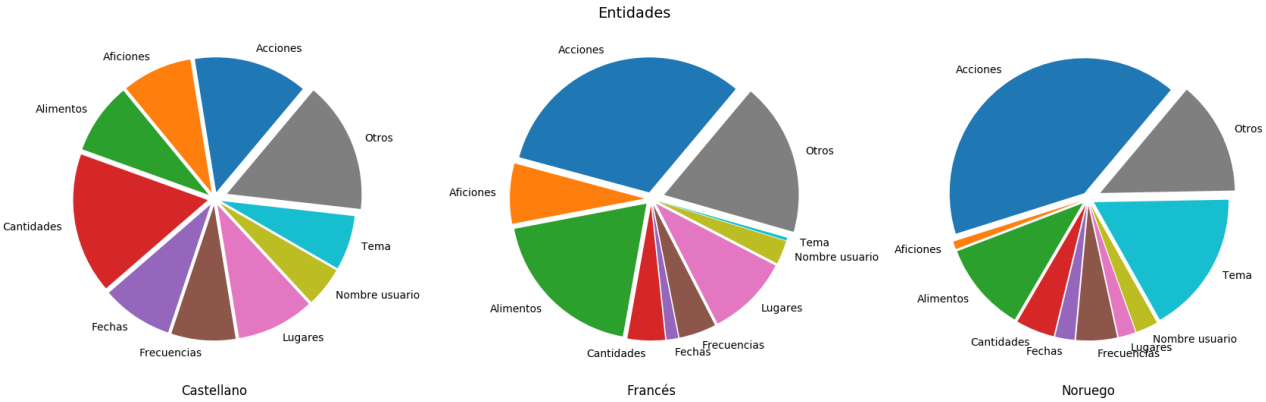

Figura 4.8: Comparativa de la distribución de las entidades en los tres idiomas.

Entidades

A nivel de entidades, lo que se puede ver en la figura 4.8 es que la entidad más utilizada es el uso de *Acciones en infinitivo*, como ya ocurría para el castellano. Sin embargo, a diferencia de lo que ocurría con el castellano, donde los modificadores de la acción (cantidades, cantidades de tiempo, frecuencias) ocupaban un segundo puesto en importancia, lo que ocurre tanto para el noruego como para el francés es que *Alimentos* toma una gran importancia, volviendo a tener lógica por la temática de las segundas sesiones. Más allá de eso, es difícil encontrar hechos destacables que se repitan, ya sea por diferencias culturales o por diferentes modos de actuar del *wizard*.

4.2. Evaluación del NLG

En esta sección se evalúan los dos NLG utilizados en este trabajo. Por un lado, se cuenta con el sistema basado en reglas implementado desde cero: GROWsetta. Mientras que por otro lado se

Nombre del módelo	Beam size	Uso de reranker	Datos utilizados
mod_1_yes_all	1 (greedy decoding)	Sí	Toda
mod_5_yes_all	5	Sí	Todos
mod_10_yes_all	10	Sí	Todos
mod_100_yes_all	100	Sí	Todos
mod_5_no_all	5	No	Todos
mod_100_no_all	100	No	Todos
mod_5_yes_own	5	Sí	Anotación propia
mod_5_yes_ann	5	Sí	Anotación anotadores

Tabla 4.3: Nombre de los experimentos realizado con TGen.

ha utilizado la herramienta de generación estadística TGen. Ante la falta de métricas automáticas para evaluar el sistema basado en reglas, en gran parte de la sección el trabajo se centra en los experimentos realizados con TGen. Una vez presentados los experimentos de TGen y evaluados con las métricas de solapamiento, se realiza una evaluación humana en la que se compara el sistema basado en reglas con cuatro de los experimentos realizados con TGen en base a diferentes criterios.

4.2.1. Introducción a los experimentos de TGen

Como ya se ha comentado a lo largo de este trabajo, TGen se ha tomado como un sistema para poder evaluar la capacidad que tienen los sistemas estadísticos para trabajar a partir de los datos con los que se contaba para esta tarea, ya que se trata de uno de los sistemas referencia en muchos artículos [22, 76, 51]. Los datos que se han utilizado en este experimento han sido únicamente los de castellano, ya que eran los más numerosos y los que por conocimiento del idioma permitían analizar su salida.

De los tres posibles algoritmos desarrollados por Dusek y Jurcicek [14, 67, 66] se ha decidido usar el sistema basado en la red seq2seq que genera frases directamente y sin uso de contexto. Por ello, para seleccionar los parámetros fijos de la red se han elegido los mismos que el artículo en el que se presenta dicha versión de TGen [67]. De este modo, como mecanismo de optimización de la red se utiliza Adam optimizer. La red que trabaja con celdas LSTM de tamaño 128, embeddings de tamaño 50, un learning rate de 0.001 y con tamaño de los batches igual a 20. Sin embargo, ante la diferencia de tamaño entre el dataset utilizado por Dusek (202) frente al que se utiliza aquí (8173)se ha decidido reducir el número de pases de los datos en la red de 1000 a 100 con una condición de parada en la que si el sistema no mejora durante 10 pases el sistema no continua entrenando. Dichos parámetros se han utilizado también para el reranker en los experimentos en los que se haya hecho uso de él. En cuanto a los parámetros del reranker se ha considerado un valor de penalización igual a 100.

Para este trabajo se han realizado 8 experimentos diferentes. Los cuatro primeros experimentos corresponden con los cuatro presentados con reranker en el artículo que se está utilizando como referencia. En dichos experimentos, como se aprecia lo único que cambia es el tamaño de secuencias que se conservan en el algoritmo de beam search, el denominado *beam size*. Posteriormente, se han realizado dos experimentos en los que se prescinde del uso del reranker para los

dos valores de beam size que mejores resultados han sacado. Finalmente, se ha decidido hacer dos experimentos con los dos datasets en los que se puede dividir la base de datos en castellano. Con todo esto, los experimentos están caracterizados en base a tres parámetros los cuales se utilizan para determinar el nombre que se le va a dar al modelo durante el trabajo (tabla 4.3): beam size, uso de reranker y el conjunto de datos utilizados.

4.2.2. Evaluación automática de los modelos de TGen

Para dichos experimentos se ha realizado un 5-cross validation haciendo 5 divisiones diferentes de los datos. En los 6 experimentos en los que se han utilizado todos los datos, las 5 divisiones para generar y evaluar el generador son siempre las mismas. Para los otros dos experimentos, como se tratan de dos datasets distintos es evidente que esas 5 divisiones no son las mismas pero sí que también se realiza la 5-cross validation. Cuando se habla de división de los datos es porque para el entrenamiento y el testeo de los datos no se han utilizado todos los datos, sino que se ha dividido el dataset total en un conjunto de entrenamiento, otro de validación y otro de test. La división realizada en todos los casos sigue una relación 3:1:1 que es la misma que utiliza Dusek et al. en un dataset con un tamaño muy similar [66].

Para la evaluación, al igual que ocurre en muchos artículos en los que se diseñan NLG se utilizan las métricas habituales basadas en solapamientos de n-gramas: BLEU, NIST, METEOR y ROUGE-L [51, 22, 76] . En algunos de estos artículos se utiliza CIDEr también, pero por su definición se ha descartado para este trabajo. A parte de dichas métricas, otra forma habitual de analizar la calidad de los generadores que parten de DA que presentan slots de atributo-valor es medir la capacidad de incluir todos ellos en el texto generado [67, 77, 28]. Para medir esa capacidad se ha utilizado la otra métrica explicada en la teoría: ERR.

Métricas basadas en solapamiento de n-gramas

Los métodos para obtener las métricas basadas en solapamiento no se han implementado manualmente, sino que se ha hecho uso de una herramienta utilizadas en el E2E challenge para evaluar los distintos generadores [*]. El hecho de que esta forma de evaluar se haya utilizado para otros generadores, permite una relativa comparación con otros generadores, ya que el dataset y el campo de aplicación no es el mismo. En el caso del E2E challenge el dataset es prácticamente 10 veces más grande que el utilizado aquí, con un número de etiquetas mucho menor y se centra en la generación de información sobre restaurantes [22], por lo que la posible comparativa que se pueda hacer no es muy concluyente.

Antes de pasar a la comparativa con otros generadores de la bibliografía, se ha tratado de encontrar la mejor configuración de TGen para el dataset con el que se ha trabajado. Lo primero que se ha hecho es analizar el uso de cuatro tamaños diferentes de beam size (1,5,10 y 100). Como se puede observar en la figura 4.9 los resultados obtenidos para los tamaños 5 y 100 destacan sobre los otros dos modelos para las cuatro métricas.

A tenor de estos resultados, se ha tratado de analizar el efecto del reranker en las distintas métricas. Para ello, lo que se ha hecho es construir modelos con los dos tamaños de beam size que mejores resultados han dado pero sin reranker. Lo que se observa (figura 4.10) es una ligera reducción de los valores en las cuatro métricas. Sin embargo, la calidad de los generadores se sigue

[*] https://github.com/tuetschek/e2e-metrics

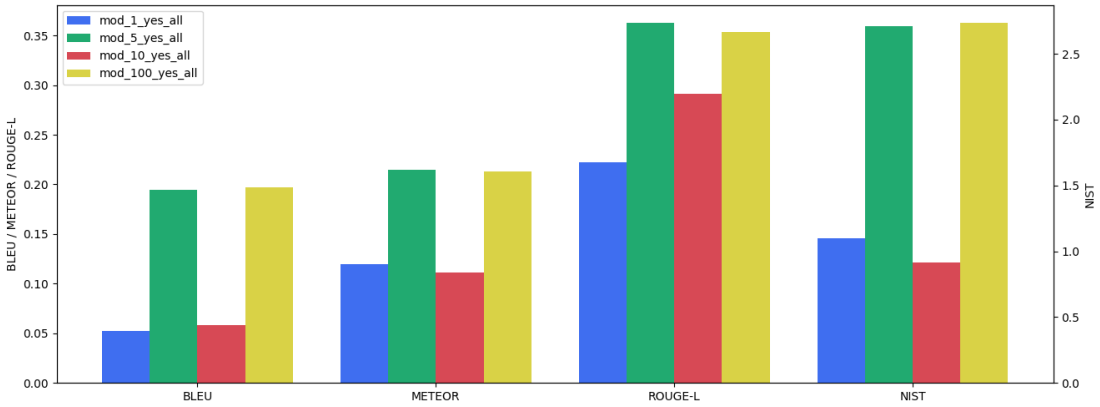

Figura 4.9: Evaluación automática de los modelos para distintos tamaños de beam size. (BLEU,METEOR y ROUGE que toman valores entre 0 y 1 se representan utilizando el eje Y del lado izquierdo; por su parte NIST no está acotado superiormente y utiliza el eje Y del lado derecho)

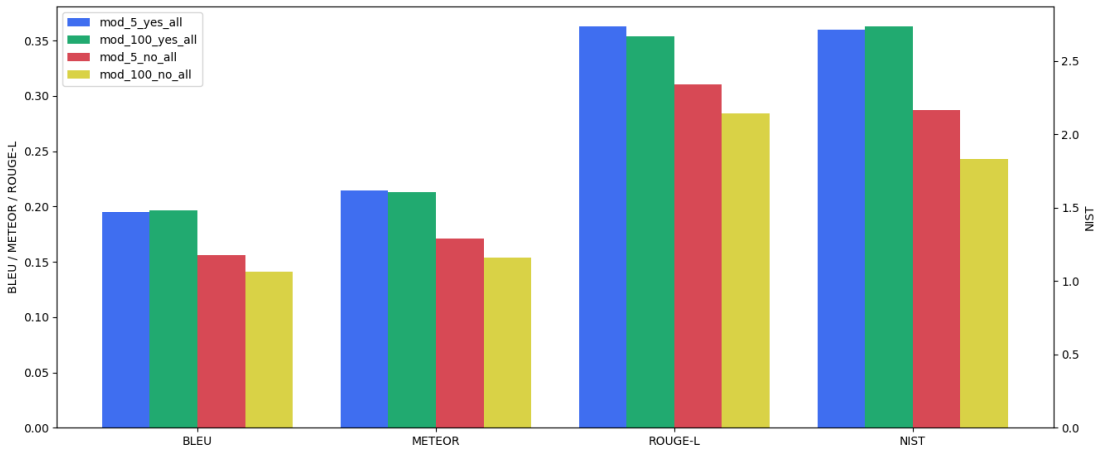

Figura 4.10: Evaluación automática de los modelos para analizar la importancia del reranker (BLEU,METEOR y ROUGE que toman valores entre 0 y 1 se representan utilizando el eje Y del lado izquierdo; por su parte NIST no está acotado superiormente y utiliza el eje Y del lado derecho).

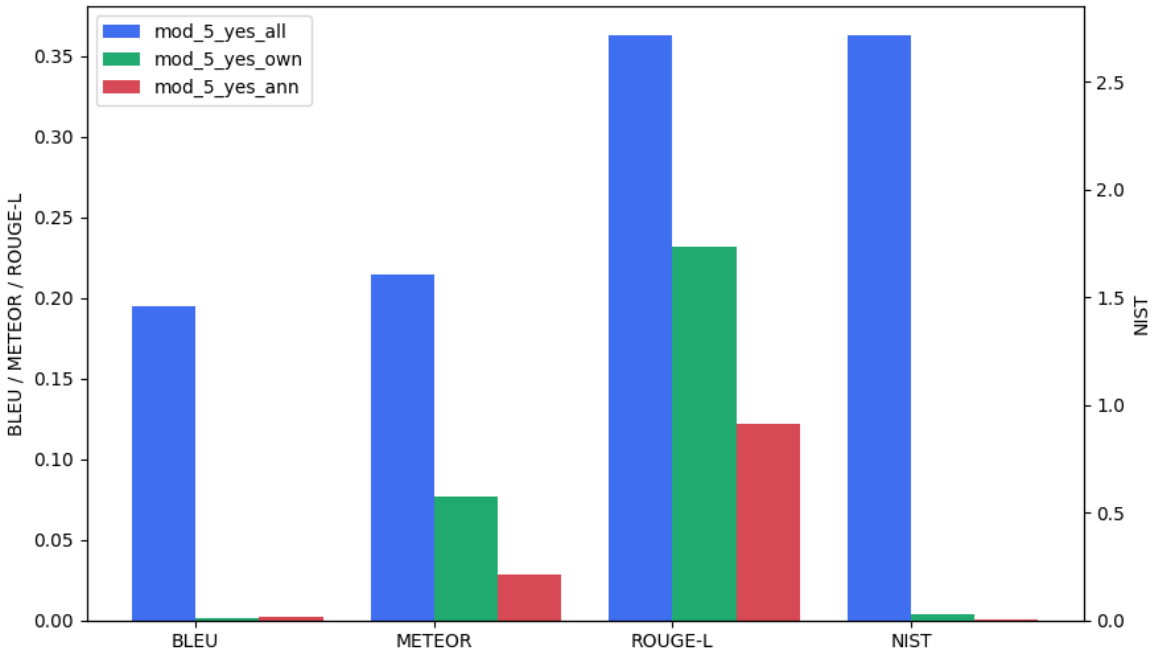

Figura 4.11: Evaluación métrica de los modelos construidos con distintos datasets (BLEU,METEOR y ROUGE que toman valores entre 0 y 1 se representan utilizando el eje Y del lado izquierdo; por su parte NIST no está acotado superiormente y utiliza el eje Y del lado derecho).

manteniendo por encima de los valores para los otros dos tamaños de beam size con reranker. Esto habla de una mayor importancia del dicho parámetro asociado al algoritmo de busqueda que la reevaluación de las frases.

Con tal de poder determinar si alguno de los datasets suponía algún tipo de déficit para la construcción de los modelos con el total de los datos, se ha realizado la construcción de dos modelos con los dos datasets de los que se dispone para el castellano. Lo que se ve claramente (figura 4.11) es que los valores para las cuatro métricas se sitúan en los valores más bajos observados. Como conclusión a estas observaciones, el uso de una gran cantidad de datos es necesario para construir un buen modelo. Por otro lado, no destaca ninguno de los dos datasets por encima del otro.

Comparando los mejores resultados obtenidos para este dataset con los que se presentan en el E2E challenge, la sensación es que aún se está lejos de un punto óptimo. Para todas las métricas, las evaluaciones automáticas obtenidas se sitúan con valores inferiores a la mitad de los mejores valores obtenidos en el challenge. Y lo que es más destacable, ninguno de los valores obtenidos en este trabajo supera ni el peor de los resultados para dichas métricas en los generadores presentados para ese reto.

Evaluación del error en la inclusión de slots

Otra forma de medir la calidad de un generador es su capacidad de incluir toda la información que se le pida que suministre. En este caso, el DA del que se parte contiene atributos con ciertos valores y lo que se va analizar en este caso es con qué frecuencia no se incluyen estos valores en las frases (se les denomina missing values, M) y en qué casos algunos valores no incluidos en el

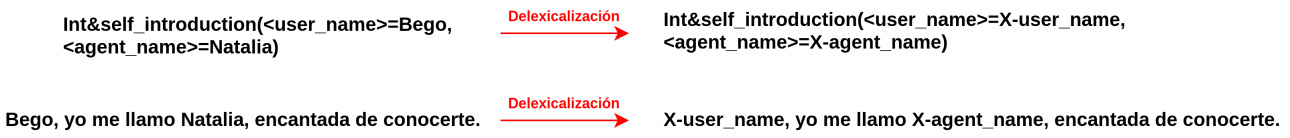

Figura 4.12: Delexicalización de un DA y una frase.

Modelo	M (ERR)	S (ERR)	ERR
mod_5_yes_all	0.608	0.743	1.351
mod_10_yes_all	0.884	0.634	1.518
mod_100_yes_all	0.598	0.700	1.298
mod_5_no_all	0.715	0.629	1.344
mod_100_no_all	0.779	0.709	1.488
mod_5_yes_own	1	0,0004	1,0004
mod_5_yes_ann	0.965	0.026	0.991

Tabla 4.4: Error en la generación de slots en tanto por uno.

DA aparecen en la frase (se les denomina superfluous values, S) sobre el total (T).

Para facilitar la búsqueda de errores con TGen, al igual que se hace en muchos trabajos con seq2seq, las frases y los DA deben ser delexicalizados, es decir, se debe asignar un valor común (X-attribute, en este caso) a todos los valores del mismo atributo (ejemplo de delexicalización en figura 4.12). De este modo, el hecho de buscar los atributos en la frase solo consiste en buscar dichos tokens con valor igual a X-attribute y ver si dichos X-attribute encontrados corresponden con los que hay en el DA.

En la tabla 4.4 se presentan los resultados del error obtenidos en tanto por uno, analizando por separado la aportación al error de los missing values y de los superfluous values. Lo que se observa es que el error para todos los modelos es muy alto (>100 % en prácticamente todos los casos) si se hace comparativa con otros generadores donde el error total se sitúan en torno al 30 % [77, 67]. Analizando la aportación de M y S al error, se puede establecer el origen del error en cada caso. Se puede ver cómo a medida que se aumenta el número de datos, se consigue reducir el número de atributos no incluidos desde un 100 % para la anotación propia a un 60 % en el mejor caso. Sin embargo, a medida que se aumenta el número de datos aparece la otra problemática posible y es que el sistema comienza a incluir datos no incluidos en el DA desde un 0 % y 2 % para los datasets reducidos, llegando a valores cercanos al 70 % para todo el dataset.

Evaluación humana

Para la evaluación humana de los generadores se ha diseñado un formulario (ver apéndice A) donde se pedía evaluar la generación realizada por cinco generadores distintos (cuatro con el estadístico TGen y otro con el basado en reglas GROWsetta). Dicha evaluación se hacía en base a los cuatro criterios que se han presentado en la sección de evaluación: corrección lingüística, naturalidad, adecuación al DA y variabilidad. Para ello, en el formulario se les daban 45 frases generadas para 15 DAs distintos, es decir, 3 frases por DA y se les pedía a los participantes que evaluasen en una escala de 1 a 5 cada uno de los criterios, tomando como referencia la forma de proceder en otros experimentos muy similares [51, 50]. Otra de de las cosas que también se suele repetir en este tipo se experimentos es determinar la preferencia sobre uno u otro generador [51, 66]. En este caso, a los participantes en la evaluación humana se les ha pedido que establezcan un ranking de preferencia entre los 5 generadores.

DA	MQ&obj_bennefits()	PFO&what_obj(<user_name>=Alain, <action>=hacer deporte frecuentemente)
mod_1_yes_all	¿qué más te beneficiaría bien?	Alain, ¿qué esperas obtener de este rato de conversación, Alain?
	¿qué más te beneficiaría bien?	¿qué te aporta dar en relación a tu objetivo de hacer deporte frecuentemente?
GROWsetta	¿para qué quieres conseguir el objetivo que has mencionado?	¿cuál es tu objetivo?
	¿y qué beneficio crees que te aportaría?	Alain, ¿con qué tiene que ver tu objetivo, entonces, de practicar actividad física frecuentemente?
mod_100_no_all	¿por qué más?	¿qué tal?
	¿por qué crees que es importante una alimentación saludable?	dime Alain, ¿cómo te gustaría hacer deporte frecuentemente?
mod_100_yes_all	¿por qué crees que es importante una alimentación saludable?	¿en qué te puedo ayudar en relación a tu objetivo?
	¿en qué te beneficiaría conseguirlo?	dime Alain, tienes alguna cosa que te gustaría conseguir o mejorar respecto a el objetivo de hacer deporte frecuentemente?
mod_5_yes_all	imagina que pierdes peso ¿qué te aportaría?	haces a ti el objetivo cuando te gustaría conseguir tu alimentación?
	¿qué beneficios obtendrías si consiguieras cambiar tu forma de cocinar de acuerdo al objetivo?	¿en qué otras cosas te veías empleando tu tiempo, Alain, cuando trabajabas?

Tabla 4.5: Ejemplos de frases generadas para los distintos DAs.

En la evaluación han contribuido 22 participantes que tenían como lengua nativa el castellano, de los cuales 16 eran hombres y 6 eran mujeres. La edad media de dichos participantes ha sido en torno a los 26 años. En cuanto a los generadores estadísticos elegidos para dicha evaluación, se han escogido los dos que mejor evaluación han dado con las métricas automáticas ($mod\_5\_yes\_all$ y $mod\_100\_yes\_all$), uno de los que peores resultado ha dado ($mod\_1\_yes\_all$) y uno de los que se ha implementado sin reranker ($mod\_100\_yes\_all$).

Antes de presentar los resultados obtenidos en dichos formularios, se va a presentar un par de ejemplos (tabla 4.5) de lo que produce cada generador de cada DA de los escogidos para el formulario. El resto se pueden ver en el apéndice A. En dicha tabla, se presentan las que se consideran peor y mejor generación de las tres generaciones hechas para cada DA (en la tabla se sitúa la peor en la fila superior de cada modelo y la mejor en la inferior para los dos DAs analizados) para poder analizar sus puntos fuertes y débiles. Al ser solo dos ejemplos es difícil sacar conclusiones muy generales con respecto a todos los criterios pero sí que se aprecia un mejor funcionamiento del sistema basado en reglas y del sistema estadístico entrenado con un tamaño 100 de beam size y haciendo uso de reranker ($mod\_100\_yes\_all$).

En la figura 4.13 se muestra la valoración media que han tenido los cinco generadores en base a los cuatro criterios. Se confirma que los dos mejores generadores son los mencionados anteriormente. Destacan sobre el resto alcanzando valores cercanos al 4 en todas las métricas salvo en variabilidad, por lo que se estaría hablando de la consideración de dos buenos generadores. En el caso de la variabilidad, el modelo estadístico sigue siendo bueno pero en el caso de GROWsetta la variabilidad es uno de sus puntos débiles, como lo es en muchas ocasiones de los sistemas basados en reglas, presentando una media que le sitúa un poco por debajo de lo que sería aceptable. Por su parte, el generador estadístico que utiliza greedy decoding como método de generación

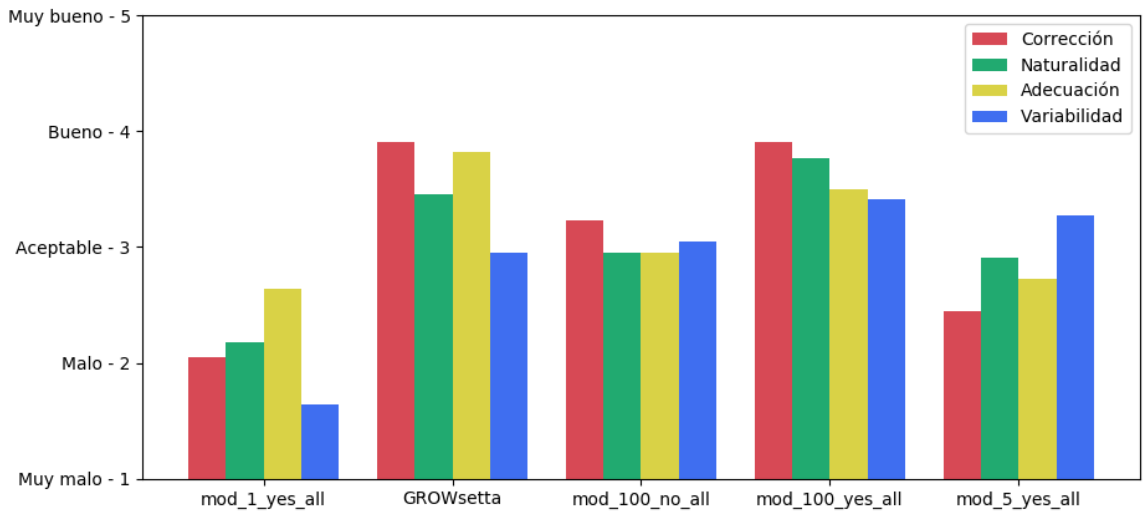

Figura 4.13: Resultados obtenidos de los cuestionarios de evaluación humana.

de las sentencias ($mod\_1\_yes\_all$) destaca como el sistema con peor calidad en base a todos los criterios, con muy poca variabilidad y problemas a nivel lingüístico lo que seguramente derive en una mala naturalidad. En cuanto a los otros dos modelos se encuentran en un punto intermedio donde todas las métricas están cerca de la aceptabilidad ya sea un poco por debajo o por encima, por lo que se puede hablar de generadores con nivel aceptable y sin grandes diferencias entre ellos salvo que $mod\_100\_no\_all$ es más correcto lingüísticamente.

Por último,a parte de evaluar los generadores en base a estas métricas se les ha pedido a los participantes hagan un ranking de los generadores. En dicha clasificación los usuarios le asignaban un 1 al mejor, un 2 al segundo y así hasta el 5 para el peor. Lo que se observa es que básicamente dicha clasificación viene a reafirmar lo obtenido con las valoraciones de los criterios. El modelo estadístico con mejor evaluación en las métricas se sitúa en primera posición con una posición media de 2.05, seguido muy cerca por GROWsetta con una posición media de 2.14. Por su parte, los dos modelos que se situaban a nivel intermedio sin grandes diferencias a nivel de ranking sí que presentan una mayor diferencia de lo esperado a tenor de lo visto con los criterios de evaluación. El modelo $mod\_100\_no\_all$ se sitúa en tercera posición (2.73 de posición media) muy distanciado del $mod\_5\_yes\_all$ (3.41 de posición media) a pesar de que en los criterios el primero de ellos solo lo superaba claramente en la corrección lingüística. Por su parte, $mod\_1\_yes\_all$ se confirma como el peor generador de los cinco al ser situado por prácticamente todos los participantes en última posición.

Comparativa entre métricas automáticas y evaluación humana

Para finalizar con la evaluación, se ha hecho una comparativa de las métricas y la evaluación humana, con el objetivo de validar cada una de ellas, ya que finalmente la evaluación que prevalece es la del humano y más en una aplicación como la de EMPATHIC. No se ha hecho una comparativa uno a uno entre las métricas y los criterios utilizados en la evaluación humana. Para recoger la evaluación humana bajo un único valor se ha hecho una media entre los valores de los siguientes criterios: corrección, naturalidad y adecuación. El hecho de que la variabilidad se quede fuera es porque por definición de las métricas automáticas utilizadas en ningún caso están relacionadas con dicho criterio. Así, se ha realizado una comparativa de cada una de las métricas automáticas con ese valor medio que define la evaluación humana.

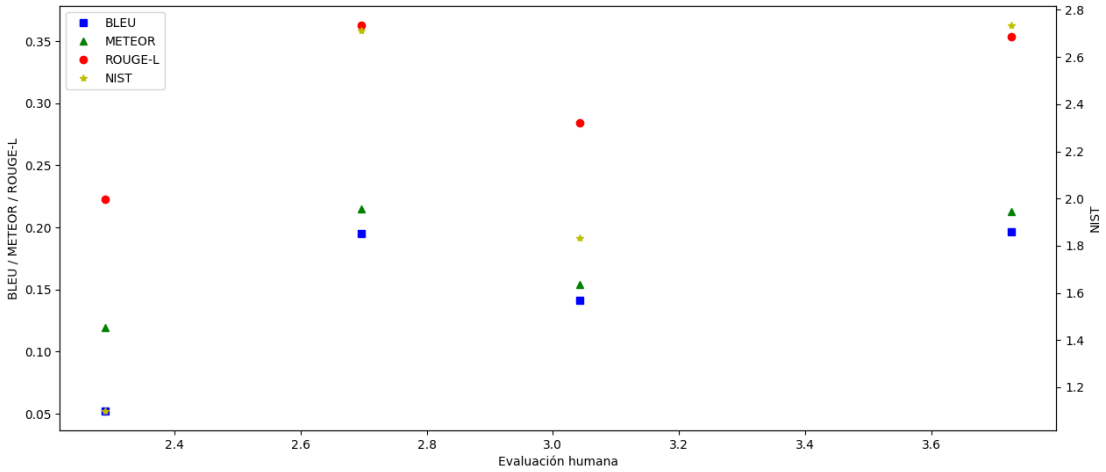

Figura 4.14: Comparativa entre las métricas automáticas y la evaluación humana (los puntos situados a la misma altura en el eje X corresponden a un mismo modelo, por ello comparten el valor asociado a la valoración humana).

En estas comparativas se debe tener en cuenta que no se ha realizado evaluación humana de todos los modelos estadísticos, por lo que solo los cuatro incluidos en dichos formularios se pueden utilizar para la comparativa. Es más, solo esos cuatro van a ser los que aparezcan en la comparativa, ya que el otro sistema que se ha evaluado bajo criterios humanos es GROWsetta, del cual no se tienen valores de las métricas automáticas.

De lo expuesto en la figura 4.14 lo que se observa que ninguna de las métricas tiene una relación directa con la evaluación humana. La falta de un gran número de generadores hace que las conclusiones que se pueden sacar no sean categóricas. Sin embargo, sí que se ve que uno de los sistemas con mayor valoración en métricas automáticas ($mod\_5\_yes\_all$) se ve superado en valoración humana por el sistema sin reranker ($mod\_100\_no\_all$) a pesar de que en métricas automáticas en todas este segundo se situaba por detrás.

Capítulo 5

Conclusiones y trabajo futuro

En esta memoria se han presentado los trabajos llevados a cabo para la implementación del EMPATHIC-NLG, un generador de lenguaje natural multilingüe (castellano, francés y noruego) que tiene como finalidad última su integración en el EMPATHIC-VC. Así, englobado dentro del proyecto EMPATHIC dicho NLG debía generar las respuesta de un sistema que simulaba la función de un coach con el objetivo de que la población de avanzada edad tenga un estilo de vida más saludable. Debido a que se trata de un dominio muy concreto y por tanto poco explorado en el campo de los NLGs, el trabajo ha presentado dos fases: definición de una base de datos e implementación de un generador adaptado a EMPATHIC.

Durante la primera fase, las dos tareas a realizar fueron definir qué tipo de dato de entrada iba a ser el de partida para la generación del texto en el NLG y en base a esto realizar el etiquetado de los datos de los que se contaba. La primera decisión fue que los DAs con los que se iban a trabajar se componían de seis elementos diferentes: etiqueta, subetiqueta, slots, polaridad, género del usuario y género del coach. Así, el proceso de anotación se realizó sabiendo que había que anotar dichos seis elementos, salvo en el caso del noruego donde debido a características del idioma la anotación de los géneros no tenía sentido. Para los tres idiomas se tomaron datos de sesiones reales de usuarios con el VC a través de una plataforma de WoZ, con el añadido de que para el castellano se contaba con otro tipo de datos obtenidos gracias a la colaboración de una coach profesional.

El resultado de la anotación deja claro que la taxonomía elegida compuesta por etiquetas y subetiquetas es bastante adecuada, ya que en las conversaciones reales se puede ver cómo prácticamente todas ellas son utilizadas. Sí que hay algunas subetiquetas cuyo uso en las sesiones reales es muy reducido o nulo pero se cree que es por la reducida duración de las sesiones, que en algunos casos no ha permitido avanzar en las fases del modelo GROW. De hecho, en base a las etiquetas utilizadas en las sesiones reales se puede ver hasta qué fase han avanzado las conversaciones. Se ve que en la gran mayoría de casos no superan la fase R de análisis de la realidad. Seguramente, si se quiere tener una base de datos para construir un sistema que sepa llevar una conversación que pase por todas las fases se deberá intentar que en las próximas sesiones que se va a tener con usuarios reales se llegue hasta el final en el modelo GROW. Otra problemática derivada de los datos es que desde EMPATHIC se ha decidido tomar como polaridad por defecto la polaridad positiva para el VC y en las anotaciones realizadas en francés y noruego (por los resultados obtenidos) da la sensación de que los anotadores han etiquetado tomando como polaridad por defecto la polaridad neutra. Esta discrepancia habrá que subsanarla en algún momento si es necesaria y posible. Por último, más allá de que las conversaciones para el noruego cuentan con un número más reducido de intervenciones por parte del coach, las diferencias en la anotación entre los distintos idiomas no son apreciables a pesar de las diferencias culturales, de que los Wizards encargados de llevar a

cabo la labor de coach eran personas diferentes y de que el etiquetado también está realizado por anotadores distintos.

En cuanto a la segunda fase, en este trabajo se ha desarrollado un sistema basado en reglas: GROWsetta y se ha hecho uso de un generador estadístico ya implementado, TGen. GROWsetta es un sistema basado en plantillas muy adaptado a la taxonomía de etiquetas elegida y que permite ir actualizándolo en base a lo que el DM pueda producir. Por su parte, TGen se ha seleccionado porque es un sistema estadístico muy adaptable a los datos de los que se partía y porque se ha visto en la bibliografía que superaba a muchos de los generadores implementados del mismo tipo.

Para la evaluación de ambos generadores en este trabajo solo se han utilizado métodos intrínsecos, tanto con métricas automáticas como con evaluación humana. En breve, desde EMPATHIC se realizarán las primeras sesiones con usuarios reales, donde se utilizará el primer prototipo con todos los módulos trabajando juntos, entre ellos el NLG, lo que va a permitir una evaluación extrínseca del NLG con la opinión de los usuarios reales sobre las respuestas como parte del EMPATHIC-VC. Hasta ahora, de la evaluación realizada con las métricas, lo primero que se observa es que la calidad de los generadores estadísticos obtenidos con nuestra base de datos está lejos de lo que consigue TGen u otros generadores similares con otras bases de datos. De aquí se sacan dos posibles formas de actuar: aumentar la cantidad de datos o variar de alguna forma la base de datos que se tiene ahora. La primera de las posibilidades, en base a lo visto, aseguraría mejores resultados ya que se ha visto en este mismo trabajo que con el mismo tipo de datos pero con un conjunto reducido de datos los resultados son aún peores. En cuanto a la posibilidad de modificar la base de datos, una de las medidas que se ha pensado que se podrían tomar es cambiar la forma de presentar la etiqueta y subetiqueta a TGen. Actualmente, para TGen todo par etiqueta-subetiqueta es igual de distinto por lo que la idea sería buscar una alternativa de presentar los datos que no haga que se pierda la taxonomía definida para las etiquetas a la hora de trabajar dentro de TGen.

Una de las cosas que se ha hecho a través de las métricas automáticas de evaluación es tratar de buscar una configuración óptima de TGen para la base de datos con la que se trabajaba. La conclusión más clara es que se necesita de un conjunto de datos más grande para que dichas métricas mejoren. La otra conclusión establecía que el parámetro asociado a la decodificación basado en beam search era más importante que el uso de un reranker. Sin embargo, esta segunda conclusión no se confirmaba al realizar la evaluación humana.

Antes de pasar a las conclusiones que deja la evaluación humana, analizamos lo obtenido del slot ERR. Lo visto con dicho error es que TGen tenía muchos problemas con la generación de slots para esta base de datos, a pesar de que el reranker se implementa con la idea de que dicho problema no apareciese. De este modo, si la idea es seguir con TGen o con otro tipo de generador E2E basado en ideas similares, será necesario incluir algún tipo de elemento que asegure la inclusión de los slots y que evite incluir los que no aparecen el DA. Dentro de la bibliografía ya se han visto algunos elementos como podían ser mecanismos de copia que podrían ser válidos. Otra idea a futuro, dentro de buscar herramientas similares a TGen pero que den mejores resultados, es utilizar la versión de TGen que hace uso del contexto.

En cuanto a la evaluación humana, lo que se observa es que dos de los generadores tienen una buena valoración, siendo uno de ellos el sistema basado en reglas GROWsetta y otro uno de los generadores diseñados con TGen, lo que deja los dos enfoques como posibles caminos a seguir en la mejora del NLG-EMPATHIC. Además, de la comparativa entre la evaluación humana y las métricas, se establece que no hay una relación directa entre ellas. De este modo, sería interesante explorar la búsqueda de una métrica de evaluación automática que casara mejor con la evalua-

ción humana, ya que esta última es siempre la más importante para los NLGs.

Finalmente, a nivel personal este trabajo me ha permitido conocer un campo de investigación nuevo sobre el que me gustaría seguir trabajando: el NLG. No se trataba de un campo totalmente desconocido al inicio del trabajo porque sí que había realizado alguna investigación previa en temas de NLP. Sin embargo, lo que sí ha cambiado, al ver sus capacidades dentro del proyecto EMPATHIC, es la sensación de utilidad de este tipo de herramientas. Por un lado, el hecho de desarrollar SDSs me parece que es una forma magnífica de acercar estos sistemas a personas menos hábiles con la informática. Y más allá de eso, al ser una herramienta capaz de servir de guía o que básicamente permita a una persona comunicarse y perder la sensación de soledad, muestran la necesidad de seguir avanzando con estas líneas de investigación debido a su posible contribución a la sociedad.

Bibliografía

[1] M. I. Torres, J. M. Olaso, C. Montenegro, R. Santana, A. Vázquez, R. Justo, J. Lozano, S. Schlögl, G. Chollet, N. Dugan, et al., The empathic project: mid-term achievements, in: Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments, ACM, 2019, pp. 629–638.

[2] A. López Zorrilla, M. de Velasco Vázquez, J. Irastorza Manso, J. M. Fernández Olaso, R. Justo Blanco, M. I. Torres Barañano, et al., Empathic: Empathic, expressive, advanced virtual coach to improve independent healthy-life-years of the elderly, Procesamiento del Lenguaje Natural 61 (2018) 167–170.

[3] L. Brinkschulte, N. Mariacher, S. Schlögl, M. I. Torres, R. Justo, J. M. Olaso, A. Esposito, G. Cordasco, G. Chollet, C. Glackin, et al., The empathic project: building an expressive, advanced virtual coach to improve independent healthy-life-years of the elderly, in: SMARTER LIVES 2018: digitalisation and quality of life in the ageing society, Universität Inssbrück, 2018, pp. 36–52.

[4] D. C. Willcox, G. Scapagnini, B. J. Willcox, Healthy aging diets other than the mediterranean: a focus on the okinawan diet, Mechanisms of ageing and development 136 (2014) 148–162.

[5] N. Davies, Promoting healthy ageing: the importance of lifestyle, Nursing Standard (through 2013) 25 (19) (2011) 43.

[6] D. Ding, H.-Y. Liu, R. Cooper, R. A. Cooper, A. Smailagic, D. Siewiorek, Virtual coach technology for supporting self-care, Physical Medicine and Rehabilitation Clinics 21 (1) (2010) 179–194.

[7] K. Cavanagh, A. Millings, Interpersonal computing: the role of the therapeutic relationship in e-mental health, Journal of Contemporary Psychotherapy 43 (4) (2013) 197–206.

[8] E. Reiter, R. Dale, Building natural language generation systems, Cambridge university press, 2000.

[9] D. J. Trzpis, Adaptación de una herramienta de generación de lenguaje natural al idioma espanol, Ph.D. thesis, ETSI_Informatica (2015).

[10] E. Manishina, Data-driven natural language generation using statistical machine translation and discriminative learning, Ph.D. thesis (2016).

[11] M. Vicente, C. Barros, F. S. Peregrino, F. Agulló, E. Lloret, La generación de lenguaje natural: análisis del estado actual, Computación y Sistemas 19 (4) (2015) 721–756.

[12] C. Montenegro, A. López Zorrilla, J. M. Olaso, R. Santana, R. Justo, J. A. Lozano, M. I. Torres, A dialogue-act taxonomy for a virtual coach designed to improve the life of elderly, Multimodal Technologies and Interaction 3 (3) (2019) 52.

[13] K. Staykova, Natural language generation and semantic technologies, Cybernetics and Information Technologies 14 (2) (2014) 3–23.

[14] O. Dušek, F. Jurcicek, Training a natural language generator from unaligned data, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2015, pp. 451–461.

[15] S. Agarwal, O. Dusek, I. Konstas, V. Rieser, Improving context modelling in multimodal dialogue generation, arXiv preprint arXiv:1810.11955.

[16] D. Marcheggiani, L. Perez-Beltrachini, Deep graph convolutional encoders for structured data to text generation, arXiv preprint arXiv:1810.09995.

[17] Q. Wang, X. Pan, L. Huang, B. Zhang, Z. Jiang, H. Ji, K. Knight, Describing a knowledge base, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 10–21.

[18] M. A. Walker, S. Whittaker, A. Stent, P. Maloor, J. D. Moore, M. Johnston, G. Vasireddy, Speechplans: Generating evaluative responses in spoken dialogue, in: Proceedings of the International Natural Language Generation Conference, 2002, pp. 73–80.

[19] J. G. C. de Souza, M. Kozielski, P. Mathur, E. Chang, M. Guerini, M. Negri, M. Turchi, E. Matusov, Generating e-commerce product titles and predicting their quality, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 233–243.

[20] V. Harrison, M. Walker, Neural generation of diverse questions using answer focus, contextual and linguistic features, arXiv preprint arXiv:1809.02637.

[21] X. Li, K. J. Van Deemter, C. Lin, Statistical nlg for generating the content and form of referring expressions, in: Proceedings of the 11th International Conference on Natural Language Generation, Association for Computational Linguistics (ACL), 2018.

[22] O. Dušek, J. Novikova, V. Rieser, Findings of the e2e nlg challenge, arXiv preprint arXiv:1810.01170.

[23] A. Gatt, E. Krahmer, Survey of the state of the art in natural language generation: Core tasks, applications and evaluation, Journal of Artificial Intelligence Research 61 (2018) 65–170.

[24] M. Madhavan, Natural language generation scope, applications and approaches.

[25] L. Bell, J. Boye, J. Gustafson, Real-time handling of fragmented utterances, in: Proc. NAACL workshop on adaptation in dialogue systems, 2001, pp. 2–8.

[26] A. Raux, B. Langner, A. W. Black, M. Eskenazi, Let's go: Improving spoken dialog systems for the elderly and non-natives, in: Eighth European Conference on Speech Communication and Technology, 2003.

[27] G. Skantze, Error handling in spoken dialogue systems-managing uncertainty, grounding and miscommunication, Gabriel Skantze, 2007.

[28] T.-H. Wen, M. Gasic, N. Mrksic, P.-H. Su, D. Vandyke, S. Young, Semantically conditioned lstm-based natural language generation for spoken dialogue systems, arXiv preprint arXiv:1508.01745.

[29] A. H. Oh, A. I. Rudnicky, Stochastic natural language generation for spoken dialog systems, Computer Speech & Language 16 (3-4) (2002) 387–407.

[30] S. Agarwal, M. Dymetman, E. Gaussier, Char2char generation with reranking for the e2e nlg challenge, arXiv preprint arXiv:1811.05826.

[31] A. Leuski, D. R. Traum, Npceditor: A tool for building question-answering characters., in: LREC, Citeseer, 2010.

[32] L. Gatti, C. van der Lee, M. Theune, Template-based multilingual football reports generation using wikidata as a knowledge base, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 183–188.

[33] J. Forrest, S. Sripada, W. Pang, G. Coghill, Towards making nlg a voice for interpretable machine learning, in: Proceedings of The 11th International Natural Language Generation Conference, Association for Computational Linguistics (ACL), 2018.

[34] N. Parde, R. Nielsen, Automatically generating questions about novel metaphors in literature, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 264–273.

[35] A. Gatt, E. Reiter, Simplenlg: A realisation engine for practical applications, in: Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009), 2009, pp. 90–93.

[36] L. Anselma, A. Mazzei, Designing and testing the messages produced by a virtual dietitian, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 244–253.

[37] K. V. Deemter, M. Theune, E. Krahmer, Real versus template-based natural language generation: A false opposition?, Computational Linguistics 31 (1) (2005) 15–24.

[38] E. Reiter, Nlg vs. templates, arXiv preprint cmp-lg/9504013.

[39] C. Barros, E. Lloret, Input seed features for guiding the generation process: A statistical approach for spanish, in: Proceedings of the 15th European Workshop on Natural Language Generation (ENLG), 2015, pp. 9–17.

[40] F. Mairesse, M. Gašić, F. Jurčíček, S. Keizer, B. Thomson, K. Yu, S. Young, Phrase-based statistical language generation using graphical models and active learning, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 1552–1561.

[41] G. Jagfeld, S. Jenne, N. T. Vu, Sequence-to-sequence models for data-to-text natural language generation: Word-vs. character-based processing and output diversity, arXiv preprint arXiv:1810.04864.

[42] S. Gehrmann, F. Z. Dai, H. Elder, A. M. Rush, End-to-end content and plan selection for data-to-text generation, arXiv preprint arXiv:1810.04700.

[43] H. Bunt, The dit++ taxonomy for functional dialogue markup, in: AAMAS 2009 Workshop, Towards a Standard Markup Language for Embodied Dialogue Acts, 2009, pp. 13–24.

[44] J. L. Austin, How to do things with words, Oxford university press, 1975.

[45] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. V. Ess-Dykema, M. Meteer, Dialogue act modeling for automatic tagging and recognition of conversational speech, Computational Linguistics 26 (3) (2000) 339–373.

[46] H. Bunt, J. Alexandersson, J. Carletta, J.-W. Choe, A. C. Fang, K. Hasida, K. Lee, V. Petukhova, A. Popescu-Belis, L. Romary, et al., Towards an iso standard for dialogue act annotation, in: Seventh conference on International Language Resources and Evaluation (LREC'10), 2010.

[47] R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. K. Joshi, B. L. Webber, The penn discourse treebank 2.0., in: LREC, Citeseer, 2008.

[48] M. G. Core, J. Allen, Coding dialogs with the damsl annotation scheme, in: AAAI fall symposium on communicative action in humans and machines, Vol. 56, Boston, MA, 1997.

[49] H. Bunt, Dynamic interpretation and dialogue theory, The structure of multimodal dialogue 2 (1999) 1–8.

[50] B. Langner, Data-driven natural language generation: Making machines talk like humans using natural corpora, School of Computer Science-Carnegie Mellon University.

[51] J. M. Deriu, M. Cieliebak, Syntactic manipulation for generating more diverse and interesting texts, in: 11th International Conference on Natural Language Generation (INLG 2018), Tilburg, The Netherlands, 05-08 November 2018, Association for Computational Linguistics, 2018, pp. 22–34.

[52] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th annual meeting on association for computational linguistics, Association for Computational Linguistics, 2002, pp. 311–318.

[53] G. Doddington, Automatic evaluation of machine translation quality using n-gram co-occurrence statistics, in: Proceedings of the second international conference on Human Language Technology Research, Morgan Kaufmann Publishers Inc., 2002, pp. 138–145.

[54] A. Lavie, A. Agarwal, Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments, in: Proceedings of the Second Workshop on Statistical Machine Translation, Association for Computational Linguistics, 2007, pp. 228–231.

[55] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, in: Text summarization branches out, 2004, pp. 74–81.

[56] M. I. Torres, J. M. Olaso, N. Glackin, R. Justo, G. Chollet, A spoken dialogue system for the empathic virtual coach, in: Proceedings of the International Workshop on Spoken Dialog System Technology (IWSDS), Singapore, 2018, pp. 14–16.

[57] V. Zue, S. Seneff, J. R. Glass, J. Polifroni, C. Pao, T. J. Hazen, L. Hetherington, Juplter: a telephone-based conversational interface for weather information, IEEE Transactions on Speech and Audio Processing 8 (1) (2000) 85–96.

[58] S. Kim, L. F. D'Haro, R. E. Banchs, J. D. Williams, M. Henderson, The Fourth Dialog State Tracking Challenge, Springer Singapore, Singapore, 2017, pp. 435–449.

[59] E. Levin, R. Pieraccini, W. Eckert, A stochastic model of human-machine interaction for learning dialog strategies, IEEE Transactions on Speech and Audio Processing 8 (1) (2000) 11–23.

[60] J. M. Olaso, P. Milhorat, J. Himmelsbach, J. Boudy, G. Chollet, S. Schlögl, M. I. Torres, A Multilingual Evaluation of the vAssist Spoken Dialog System. Comparing Disco and RavenClaw, Springer Singapore, Singapore, 2017, pp. 221–232.

[61] A. Graham, Behavioural coaching - the GROW model, in: Passmore, Jonathan. Excellence in coaching: the industry guide (2nd ed.), 2006, pp. 83–93.

[62] R. C. Davis, T. S. Saponas, M. Shilman, J. A. Landay, Sketchwizard: Wizard of oz prototyping of pen-based user interfaces, in: Proceedings of the 20th annual ACM symposium on User interface software and technology, ACM, 2007, pp. 119–128.

[63] A. Fiedler, M. Gabsdil, H. Horacek, A tool for supporting progressive refinement of wizard-of-oz experiments in natural language, in: International conference on intelligent tutoring systems, Springer, 2004, pp. 325–335.

[64] C. D. Hundhausen, A. Balkar, M. Nuur, S. Trent, Woz pro: a pen-based low fidelity prototyping environment to support wizard of oz studies, in: CHI'07 Extended Abstracts on Human Factors in Computing Systems, ACM, 2007, pp. 2453–2458.

[65] S. Schlögl, G. Doherty, N. Karamanis, S. Luz, Webwoz: A wizard of oz prototyping framework, 2010, pp. 109–114.

[66] O. Dušek, F. Jurcıcek, A context-aware natural language generation dataset for dialogue systems, in: Workshop on Collecting and Generating Resources for Chatbots and Conversational Agents-Development and Evaluation, 2016, pp. 6–9.

[67] O. Dušek, F. Jurčíček, Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings, arXiv preprint arXiv:1606.05491.

[68] I. Konstas, M. Lapata, A global model for concept-to-text generation, Journal of Artificial Intelligence Research 48 (2013) 305–346.

[69] P. E. Hart, N. J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE transactions on Systems Science and Cybernetics 4 (2) (1968) 100–107.

[70] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, Journal of machine learning research 3 (Feb) (2003) 1137–1155.

[71] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.

[72] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078.

[73] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Advances in neural information processing systems, 2014, pp. 3104–3112.

[74] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, Journal of Machine Learning Research 12 (Jul) (2011) 2121–2159.

[75] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[76] Y. Puzikov, I. Gurevych, E2e nlg challenge: Neural models vs. templates, in: Proceedings of the 11th International Conference on Natural Language Generation, 2018, pp. 463–471.

[77] G. Lampouras, A. Vlachos, Imitation learning for language generation from unaligned data, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 1101–1112.

Apéndice A

Documento de evaluación humana de los generadores

En este apéndice se presenta el formulario que se le ha entregado a los participantes en el proceso de evaluación humana. A ellos no se les ha indicado a qué tipo de generador correspondía cada modelo, pero para una mejor comprensión y posible análisis del lector, en la tabla A.1 se define la correspondencia entre los modelos del formulario y el nombre asignado a lo largo del trabajo.

Modelo A	**mod_1_yes_all**
Modelo B	**GROWsetta**
Modelo C	**mod_100_no_all**
Modelo D	**mod_100_yes_all**
Modelo E	**mod_5_yes_all**

Tabla A.1: Correspondencia entre los modelos del formulario y del trabajo.

EVALUACIÓN DE EGENRADORES DE LENGUAJES PARA EL PROYECTO EMPATHIC

En el proyecto EMPATHIC desarrollamos agentes virtuales para interaccionar con personas. Estos agentes son visuales, es decir, son modelos 3D *pseudohumanos*, que deben hablar con corrección y sentido. Las conversaciones en nuestro proyecto simulan, hasta cierto punto, las sesiones de un *coach* de bienestar. Este *coach* se dirige a personas mayores de 65 años y su objetivo general es lograr que el usuario cambie alguna pauta de comportamiento no saludable, principalmente relacionada con la alimentación. Para ello, el agente virtual sigue un plan a desarrollar en fases. En primer lugar se intenta que el usuario fije qué quiere cambiar, un objetivo, *comer menos sal* por ejemplo. Después el agente intenta explorar la realidad actual del usuario y cuanto de lejos está de alcanzar su objetivo. A continuación analiza posibles obstáculos y establece posibles opciones de acción que se ajustan a la realidad del usuario. Finalmente se define un plan de acción para alcanzar ese cambio de pauta u objetivo (modelo GROW).

Uno de los módulos más importantes del proyecto es el **módulo de generación de lenguaje natural**, que es el que **queremos evaluar**. Este módulo se encarga de generar las frases que el agente dice. El módulo conoce el tipo de frase que debe generar: frase de saludo, frase de despedida, pregunta sobre el objetivo, sobre los obstáculos, etc. En ocasiones también conoce algún atributo específico, como el nombre de la persona a saludar, nombre de enfermedad, o alguna indicación temporal concreta. Con esta información deberá generar frases de calidad, desde el punto de vista lingüístico, que se ajusten a los prerrequisitos (tipo de frase y atributos) y que sean naturales. Además el generador debe ser capaz de generar frases diferentes para las mismas precondiciones, es decir, es bueno que no diga siempre lo mismo en las mismas situaciones.

En este contexto os pedimos vuestra **colaboración para evaluar 5 generadores automáticos de lenguaje** natural basados en metodologías y de características distintas. Para ello, se os pide analizar sus capacidades a la hora de generar 15 tipos distintos de frases. En cada tipo de frase se especificará tanto el contenido que se espera de la frase así como una serie de elementos o atributos que deberían estar incluidos dentro de la frase generada. Por ejemplo:

Tipo de frase: Saludo a un usuario (nombre de usuario=Alain)
Opción 1: ¡Hola, Alain!
Opción 2: Buenos días.
Opción 3: Bienvenido, Alain.

En este caso, el tipo de frase es *Saludo a un usuario* y como información adicional se incluye que el nombre del usuario es Alain. Por cada tipo de frase se muestran tres frases generadas durante tres ejecuciones distintas del mismo generador. Las tres frases pueden ser la misma, indicando poca capacidad de variación, o bien diferentes, como en el ejemplo. En este caso las tres frases tienen sentido, pero la segunda se ajusta menos a lo que se pide ya que no incluye el nombre de usuario proporcionado.

Lo que se pide es una evaluación global de cada generador. La realización de anotaciones en cada frase puede ayudar a algunos evaluadores, pero solo se tendrá en cuanta la evaluación final global del método. Ésa se realizará en base a cuatro criterios que se deben evaluar en una escala de 1 a 5:

muy malo (1), malo (2), suficiente (3), bueno (4) y muy bueno (5). Dichos criterios son los siguientes:

- **Corrección/estilo:** Se evalúa si a nivel lingüístico las frases formadas son correctas así como su calidad a nivel de estilo, con independencia de que se ajusten o no al tipo de frase que debe generarse.
- **Naturalidad:** Determina si las frases generadas se asemejan a la forma de expresarse de un humano.
- **Adecuación/relevancia:** Mide lo adecuada que es la frase para el tipo de frase que se proporciona y el contenido/atributo que debe incluirse, en su caso. Cuanto más general sea la frase más correcta es, es decir, no debe incluir contenidos/atributos no especificados en el tipo de frase.
- **Variabilidad:** Establece la capacidad del generador de crear frases distintas (con sentido) para el mismo tipo de frase y contenido. Si las tres frases generadas son iguales la variabilidad es muy mala.

Se le asignará una nota **a cada generador** para cada uno de los criterios y al final del todo, tras haber evaluado los generadores, se asignará una posición de preferencia a cada generador. En primera posición el que mejor te parezca, en segunda posición el siguiente y así hasta el peor, que estará en la última posición.

Muchas gracias por tu colaboración.

Alain Vázquez
Speech Interactive Research Group
Depto Electricidad y Electrónica
Universidad del País Vasco
alain.vazquez@ehu.eus

Generador A

Frase 1

Pregunta sobre si el usuario sufre alguna enfermedad (enfermedad=colesterol).
¿cuánto alterada está el colesterol?
¿qué te aporta la atención?
¿qué te aporta la atención?

Frase 2

Expresar una opinión positiva.
bien.
bien.
bien.

Frase 3

Preguntar por el nombre del usuario.
¿qué tal?
¿cómo te llamas?
¿cómo te llamas?

Frase 4

Expresar duda.
no creo.
no creo.
no creo.

Frase 5

Cerrar un tema de conversación.
hemos terminado.
hemos terminado.
hemos terminado.

Frase 6

Preguntar sobre la realidad actual del usuario (fecha=actualmente)
¿qué sucede con actualmente?
¿qué sucede con la familia?
¿qué sucede con la familia?

Frase 7

Preguntar sobre los beneficios de alcanzar el objetivo.
¿qué más te beneficiaría bien?
¿qué más te beneficiaría bien?
¿qué más te beneficiaría bien?

Frase 8

Preguntar sobre la realidad actual del usuario (comida=chocolate)
y chocolate comes chocolate?
te gustan los comidas?
te gustan los comidas?

Frase 9

Preguntar sobre el cuándo a la hora de definir el plan de acción (acción=cocinar; acción=tener los primeros platos hechos)
¿cuándo vas a cocinar?
¿a qué hora vas a emplear que te acercas al 10, me llamo conversaciones de cocinar?
¿cuándo vas a cocinar?

Frase 10

Saludar al usuario (nombre de usuario=Alain)
¿qué hay de nuevo Alain?
¿qué te aporta la información en relación al objetivo de cocinar?
¿qué hay de nuevo Alain?

Frase 11

Pedir perdón.
perdona que sea un poco cotilla.
perdona que sea un poco cotilla.
perdona que sea un poco cotilla.

Frase 12

Preguntar sobre lo que le va a acercar cierta acción al objetivo (acción=tener planificada tu comida)
¿en qué medida te acerca a tu objetivo de tener planificada tu comida?
¿qué te aporta si te ha ido con tu objetivo de tener planificada tu comida?
¿en qué medida te acerca a tu objetivo de tener planificada tu comida?

Frase 13

El sistema se presenta como un sistema de coaching virtual.
soy un poco y en fase de desarrollo.
soy un poco y en fase de desarrollo.
soy un poco y en fase de desarrollo.

Frase 14

Se le pregunta al usuario del plan definido qué ha conseguido llegar a hacer.
¿qué más te ha aportado información?
¿qué más te ha aportado información?
¿qué más te ha aportado información?

Frase 15

Pregunta para definir cuál va a ser el objetivo (acción=hacer deporte frecuentemente; nombre de usuario=Alain)
Alain, ¿qué esperas obtener de este rato de conversación, Alain?
¿qué te aporta dar en relación a tu objetivo de hacer deporte frecuentemente?
¿qué te aporta dar en relación a tu objetivo de hacer deporte frecuentemente?

EVALUACIÓN DEL GENERADOR A

	1 *(muy malo)*	2 *(malo)*	3 *(aceptable)*	4 *(bueno)*	5 *(muy bueno)*
Corrección					
Naturalidad					
Adecuación					
Variabilidad					

GENERADOR B

Frase 1

Pregunta sobre si el usuario sufre alguna enfermedad (enfermedad=colesterol).
¿qué sucede concretamente con colesterol?
¿qué sucede concretamente con colesterol?
¿tienes problemas de salud?

Frase 2

Expresar una opinión positiva.
ah bien.
¡fenomenal!
me gusta.

Frase 3

Preguntar por el nombre del usuario.
¿cómo te llamas?
¿cómo te llamas?
¿cómo te llamas?

Frase 4

Expresar duda.
quizás.
quizás.
quizás.

Frase 5

Cerrar un tema de conversación.
hemos acabado la sesión.
hemos acabado la sesión.
hemos acabado la sesión.

Frase 6

Preguntar sobre la realidad actual del usuario (fecha=actualmente)
¿cuál es exactamente tu situación actual?
¿qué sucede ahora actualmente?
¿qué sucede ahora actualmente?

Frase 7

Preguntar sobre los beneficios de alcanzar el objetivo.
¿para qué quieres conseguir el objetivo que has mencionado?
¿para qué quieres conseguir el objetivo que has mencionado?
¿y qué beneficio crees que te aportaría?

Frase 8

Preguntar sobre la realidad actual del usuario (comida=chocolate)
¿cuál es exactamente tu situación actual?
¿cuál es tu situación ahora con respecto a chocolate?
¿cuál es tu situación ahora con respecto a chocolate?

Frase 9

Preguntar sobre el cuándo a la hora de definir el plan de acción (acción=cocinar; acción=tener los primeros platos hechos)
¿cuándo lo vas a hacer?
¿y cuándo te vas a cocinar?
¿cuándo lo vas a hacer?

Frase 10

Saludar al usuario (nombre de usuario=Alain)
bienvenido, Alain.
¡hola Alain!
¡hola, hola!

Frase 11

Pedir perdón.
lo siento mucho.
lo siento mucho.
lo siento mucho.

Frase 12

Preguntar sobre lo que le va a acercar cierta acción al objetivo (acción=tener planificada tu comida)
¿en qué grado te aproximaría eso a tu objetivo?
¿en qué medida tener planificada tu comida te aproxima a conseguir el objetivo?
¿en qué medida tener planificada tu comida te aproxima a conseguir el objetivo?

Frase 13

El sistema se presenta como un sistema de coaching virtual.
soy un sistema que está aprendiendo labores de coaching.
soy una asistente virtual en fase de desarrollo.
soy un sistema que está aprendiendo labores de coaching.

Frase 14

Se le pregunta al usuario del plan definido qué ha conseguido llegar a hacer.
¿qué más ha sucedido?
¿qué es lo que te ha ido bien?
¿qué avance has logrado?

Frase 15

Pregunta para definir cuál va a ser el objetivo (acción=hacer deporte frecuentemente; nombre de usuario=Alain)
entonces, ¿tu objetivo sería realizar ejercicio con frecuencia?
¿cuál es tu objetivo?
Alain, ¿con qué tiene que ver tu objetivo, entonces, de practicar actividad física frecuentemente?

EVALUACIÓN DEL GENERADOR B

	1 *(muy malo)*	2 *(malo)*	3 *(aceptable)*	4 *(bueno)*	5 *(muy bueno)*
Corrección					
Naturalidad					
Adecuación					
Variabilidad					

GENERADOR C

Frase 1

Pregunta sobre si el usuario sufre alguna enfermedad (enfermedad=colesterol).
¿tienes colesterol?
¿me puedes contar tu rutina habitual de comidas?
¿cuánto de alterada está el colesterol?

Frase 2

Expresar una opinión positiva.
bueno ha sido un placer hablar contigo.
muy bien.
bien.

Frase 3

Preguntar por el nombre del usuario.
¿qué te aportaría?
¿cómo te llamas?
¿cómo te llamas?

Frase 4

Expresar duda.
podría estar equivocada.
podría estar equivocada.
podría estar equivocada.

Frase 5

Cerrar un tema de conversación.
bueno.
hemos terminado.
hemos terminado.

Frase 6

Preguntar sobre la realidad actual del usuario (fecha=actualmente)
¿qué te han enseñado esas habilidades actualmente?
¿me puedes contar tu rutina habitual de comidas?
¿qué está ocurriendo ahora en relación a tu objetivo?

Frase 7

Preguntar sobre los beneficios de alcanzar el objetivo.
¿para qué más?
¿por qué más?
¿por qué crees que es importante una alimentación saludable?

Frase 8

Preguntar sobre la realidad actual del usuario (comida=chocolate)
¿bebes chocolate?
¿cómo te llamas?
¿me puedes contar tu rutina habitual de comidas?

Frase 9

Preguntar sobre el cuándo a la hora de definir el plan de acción (acción=cocinar; acción=tener los primeros platos hechos)
¿cuándo cocinas, vas a tener los primeros platos hechos?
¿que es importante cocinar?
¿cuándo cocinas, vas a tener los primeros platos hechos?

Frase 10

Saludar al usuario (nombre de usuario=Alain)
¡hola Alain!
¿qué hay de nuevo Alain?
¿qué tal?

Frase 11

Pedir perdón.
disculpa.
perdona que sea un poco cotilla.
perdona que sea un poco cotilla.

Frase 12

Preguntar sobre lo que le va a acercar cierta acción al objetivo (acción=tener planificada tu comida)
¿en qué medida tener planficada tu comida te acerca a tu objetivo de tener planficada tu comida?
¿qué te aportaría tener planficada tu comida?
de acuerdo, tener planficada tu comida.

Frase 13

El sistema se presenta como un sistema de coaching virtual.
si no te lo he comentado ya, estoy aprendiendo labores de coaching
si no te lo he comentado ya, estoy aprendiendo labores de coaching, pero antes de ponernos con esto, me gustaría conocerte un poco más.
si no te lo he comentado ya, estoy aprendiendo labores de coaching

Frase 14

Se le pregunta al usuario del plan definido qué ha conseguido llegar a hacer.
¿cómo te ha ido bien?
¿qué ha sucedido al realizarlo?
¿cómo te ha ido el tema?

Frase 15

Pregunta para definir cuál va a ser el objetivo (acción=hacer deporte frecuentemente; nombre de usuario=Alain)
dime Alain, ¿cómo te gustaría hacer deporte frecuentemente?
¿qué tal?
dime Alain, te gustaría hacer deporte frecuentemente o hacer deporte frecuentemente?

EVALUACIÓN DEL GENERADOR C

	1 *(muy malo)*	2 *(malo)*	3 *(aceptable)*	4 *(bueno)*	5 *(muy bueno)*
Corrección					
Naturalidad					
Adecuación					
Variabilidad					

GENERADOR D

Frase 1

Pregunta sobre si el usuario sufre alguna enfermedad (enfermedad=colesterol).
¿tienes colesterol?
¿cuáles son esos?
¿tienes colesterol?

Frase 2

Expresar una opinión positiva.
perfecto.
bien.
nos lo hemos pasado bien, ¿no?

Frase 3

Preguntar por el nombre del usuario.
te gustan?
¿cómo te llamas?
¿cómo te llamas?

Frase 4

Expresar duda.
¡no creo!
podría ser.
quizás.

Frase 5

Cerrar un tema de conversación.
bueno.
concluimos.
bueno hemos terminado.

Frase 6

Preguntar sobre la realidad actual del usuario (fecha=actualmente)
¿qué sucede ahora actualmente?
¿qué más te sucede con respecto a tu alimentación?
¿qué te sucede con respecto a tu objetivo?

Frase 7

Preguntar sobre los beneficios de alcanzar el objetivo.
¿por qué crees que es importante una alimentación saludable?
¿en qué te beneficiaría conseguirlo?
¿que te aporta?

Frase 8

Preguntar sobre la realidad actual del usuario (comida=chocolate)
¿comes chocolate?
¿qué te sucede ahora con tu objetivo?
¿qué más cosas actualmente realizas actualmente?

Frase 9

Preguntar sobre el cuándo a la hora de definir el plan de acción (accion=cocinar; acción=tener los primeros platos hechos)
¿cuándo vas a cocinar?
¿en cuánto tiempo te gustaría conseguir llegar a tu alimentación?
¿cuándo vas a cocinar?

Frase 10

Saludar al usuario (nombre de usuario=Alain)
¡hola Alain!
buenos días.
buenos días.

Frase 11

Pedir perdón.
perdona que sea un poco cotilla.
disculpa.
lo siento mucho.

Frase 12

Preguntar sobre lo que le va a acercar cierta acción al objetivo (acción=tener planificada tu comida)
¿en qué medida te va a ayudar para conseguir tu objetivo de tener planificada tu comida?
¿en qué medida te acerca esto a conseguir tu objetivo troncal?
¿en qué medida este plan te va a ayudar a tu objetivo troncal?

Frase 13

El sistema se presenta como un sistema de coaching virtual.
si no te lo he comentado ya, estoy aprendiendo labores de coaching.
si no te lo he comentado ya, estoy aprendiendo labores de coaching.
si no te lo he comentado ya, estoy aprendiendo labores de coaching.

Frase 14

Se le pregunta al usuario del plan definido qué ha conseguido llegar a hacer.
¿qué ha pasado?
¿qué te ha pasado?
¿qué te ha pasado?

Frase 15

Pregunta para definir cuál va a ser el objetivo (acción=hacer deporte frecuentemente; nombre de usuario=Alain)
dime Alain, tienes alguna cosa que te gustaría conseguir o mejorar respecto a el objetivo de hacer deporte frecuentemente?
Alain. ¿cómo te puedo ayudar?
¿en qué te puedo ayudar en relación a tu objetivo?

EVALUACIÓN DEL GENERADOR D

	1 *(muy malo)*	2 *(malo)*	3 *(aceptable)*	4 *(bueno)*	5 *(muy bueno)*
Corrección					
Naturalidad					
Adecuación					
Variabilidad					

GENERADOR E

Frase 1

Pregunta sobre si el usuario sufre alguna enfermedad (enfermedad=colesterol).
¿tienes colesterol?
haces a bicicleta o eso?
haces a el sentido?

Frase 2

Expresar una opinión positiva.
que bien.
bien.
muy interesante.

Frase 3

Preguntar por el nombre del usuario.
¿y qué más?
¿cómo te llamas?
¿cómo te llamas?

Frase 4

Expresar duda.
ah.
no estoy seguro.
podría estar equivocada.

Frase 5

Cerrar un tema de conversación.
bueno.
otro tema.
hemos terminado.

Frase 6

Preguntar sobre la realidad actual del usuario (fecha=actualmente)
¿qué te han enseñado esas habilidades actualmente?
haces en general te parece la comida de casa?
¿que haces actualmente para controlar el colesterol?

Frase 7

Preguntar sobre los beneficios de alcanzar el objetivo.
¿qué beneficios obtendrías si consiguieras cambiar tu forma de cocinar de acuerdo al objetivo?
imagina que pierdes peso ¿qué te aportaría?
¿por qué crees que es importante una alimentación saludable?

Frase 8

Preguntar sobre la realidad actual del usuario (comida=chocolate)
¿bebes chocolate?
sueles andar te han las cada de cosas de cada cosas son prioritarios.
¿comes chocolate?

Frase 9

Preguntar sobre el cuándo a la hora de definir el plan de acción (accion=cocinar; acción=tener los primeros platos hechos)
¿cuándo vas a cocinar?
¿qué te gusta esa información sobre el objetivo de mejora?
¿cuándo vas a cocinar?

Frase 10

Saludar al usuario (nombre de usuario=Alain)
hola Alain
¿haces a la alimentación?
hola Alain

Frase 11

Pedir perdón.
disculpa, referente a tu objetivo.
perdona que sea un poco cotilla.
disculpa decías.

Frase 12

Preguntar sobre lo que le va a acercar cierta acción al objetivo (acción=tener planificada tu comida)
¿qué te aportaría tener planificada tu comida?
¿qué te han ultimo conoces?
¿qué te aportaría tener planificada tu comida?

Frase 13

El sistema se presenta como un sistema de coaching virtual.
estoy aprendiendo a hablar con la gente.
si no te lo he comentado ya, estoy aprendiendo labores de coaching .
si no te lo he comentado ya, estoy aprendiendo labores de coaching .

Frase 14

Se le pregunta al usuario del plan definido qué ha conseguido llegar a hacer.
¿qué avance has logrado?
¿qué te ha aportado este primer paso en relación al objetivo?
¿qué ha sucedido al realizarlo?

Frase 15

Pregunta para definir cuál va a ser el objetivo (acción=hacer deporte frecuentemente; nombre de usuario=Alain)
¿en qué otras cosas te veías empleando tu tiempo, Alain, cuando trabajabas?
haces a ti el objetivo cuando te gustaría conseguir tu alimentación?
¿en qué otras cosas te veías empleando tu tiempo, Alain, cuando trabajabas?

EVALUACIÓN DEL GENERADOR E

	1 *(muy malo)*	2 *(malo)*	3 *(aceptable)*	4 *(bueno)*	5 *(muy bueno)*
Corrección					
Naturalidad					
Adecuación					
Variabilidad					

EVALUACIÓN COMPARATIVA DE LOS CINCO GENERADORES: A, B, C, D y E

Ordenar por calidad los generadores

ORDENACIÓN	Letra identificadora del GENERADOR
PRIMERO *(el mejor)*	
SEGUNDO	
TERCERO	
CUARTO	
QUINTO *(el peor)*	

Solo un par de datos tuyos:

Edad:
Hombre (H) o Mujer (M):

¡Gracias por tu colaboración!