REVISTA
TIA
- Revista TIA - Tecnología, Investigación y Academia -
Publicación Facultad de Ingeniería y Red de Investigaciones de Tecnología Avanzada - RITA

Procesamiento del lenguaje natural (PLN) - GPT-3, y su aplicación en la Ingeniería de Software

Autor (es): Nestor Camilo Beltran, Edda Camila Rodríguez

Citar: Néstor Camilo Beltrán, Edda Camila Rodríguez (2021), Procesamiento del lenguaje natural (PLN) - GPT-3, y su aplicación en la Ingeniería de Software. Tecnol.Investig.Academia TIA , 8(1), pp: 7-21

Procesamiento del lenguaje natural (PLN) - GPT-3, y su aplicación en la Ingeniería de Software

N. Camilo Beltrán[1], E. Camila Rodríguez[2]

Resumen: Los cambios tecnológicos han permitido que el Lenguaje Natural (LN de ahora en adelante), en el que hablan los seres humanos cotidianamente se pueda incorporar en los computadores para la realización de diversas tareas que ayudan al ser humano a una mejor redacción o comprensión de un texto extenso, traducción y otras. Sin embargo, lo que se estudia en el presente documento es la potencial aplicación en el campo de la Ingeniería de Software de modelos como GPT-3, BERT con su arquitectura Transformer y otros, además de las aplicaciones derivadas que nacen de estas como los chatbots y su relación con procesos propios del ciclo de vida del software. En este contexto, se explican las generalidades del LN, del LN Computarizado, Inteligencia Artificial, Redes Neuronales para exponer los modelos sobresalientes en el campo del Procesamiento de LN, las aplicaciones y finalmente la cercanía con la Ingeniería de Software para evaluar las perspectivas de este tipo de innovaciones.

Palabras clave: Procesamiento del Lenguaje Natural Computarizado, Ingeniería de Software, Inteligencia Artificial, GPT-3, BERT, Transformers.

Abstract.
Technological changes have allowed Natural Language (LN from now on), in which human beings speak on a daily basis, to be incorporated into computers to carry out various tasks that help human beings to better write or understand a text. long text, translation, and others. However, what is studied in this document is the potential application

1 Autor 1: Nestor Camilo Beltrán Beltrán; Ingeniero Mecatrónico, Universidad Militar; Estudiante especialización Ingeniería de Software, Universidad Distrital UDFJC; Correo: ncbeltranb@correo.udistrital.edu.co

2 Autor 2: Edda Camila Rodriguez Mojica; Ingeniera Catastral y Geodesta, Universidad Distrital UDFJC; Estudiante especialización Ingeniería de Software, Universidad Distrital UDFJC; Correo: ecrodriguezm@ correo.udistrital.edu.co

Artículo de investigación

Fecha de recepción: 2019-11-30
Fecha de aceptación: 2020-03-30

ISSN: 2344-8288 Vol. 8 No. 1
2020 Bogotá-Colombia

in the field of Software Engineering of models such as GPT-3, BERT with its Transformer architecture and others, in addition to the derivative applications that arise from these such as chatbots and its relationship with processes of the software life cycle. In this context, the generalities of LN, Computerized LN, Artificial Intelligence, Neural Networks are explained to expose the outstanding models in LN Processing, applications and finally the proximity with Software Engineering to evaluate the perspectives of this type of innovations.

Key Word: Computerized Natural Language Processing, Software Engineering, Artificial Intelligence, GPT-3, BERT, Transformers.

I. Introducción

En la última década la Inteligencia Artificial se ha convertido en el campo líder de tareas de procesamiento y generación de información a través de la aparición del aprendizaje automático o Machine Learning basado en redes neuronales (RN), que ha llevado su aplicabilidad a distintos campos como la robótica, procesamiento de voz, visión artificial, procesamiento natural del lenguaje, pretendiendo en cierta medida dotar a sistemas computacionales de la capacidad de aprender. A mitad de este año apareció una herramienta que ha sorprendido por su capacidad de realizar otras tareas para las cuales no fue diseñada, dicha herramienta se llama GPT-3 [1], esta herramienta es un modelo entrenado para generar texto a partir de un entrenamiento no supervisado, su acceso al modelo completo es restringido, pero se puede acceder a una versión Beta [2] [3].

Se plantea entonces, la comprensión general de este modelo y otros (como BERT) que han sido asimilados con muy buenos resultados en algunos procedimientos, como el análisis de sentimientos [4]. Esto para considerar las aplicaciones e identificar las posibilidades, potencialidades y debilidades que se podrían presentar en una implementación de esta tecnología para el campo de la ingeniería de software.

En el campo de la ingeniería de Software, el Procesamiento Natural del Lenguaje (de aquí en adelante PNL) aplica a la búsqueda de objetividad en las técnicas de levantamiento de requerimientos e incluso en la generación de código que apoyaría al programador, permitiendo que la escritura de código se generará de forma automática presentando unas reglas iniciales. Esto para concluir sobre las posibilidades en el campo de la Ingeniería de Software alrededor de este campo tan amplio, que evoluciona aceleradamente.

II. Contenido

A. Procesamiento del Lenguaje Natural (PLN)

Se determina al lenguaje como un medio por el cual los humanos logran comunicarse y expresar racionamiento, este medio está sustentado por la asociación de signos con ciertos significados [5] .El lenguaje usa herramientas como la escritura, las señales y la voz para establecer comunicación, aquí es donde se encuentran dos tipos de lenguaje, el lenguaje natural donde encontramos lo que comúnmente llamamos idiomas como inglés, español, alemán entre otros, estos lenguajes están en constante crecimiento sin tener en cuenta las reglas que los suceden; Por otro lado tenemos los lenguajes formales que se encuentran enmarcados dentro de algunos campos como la matemática, la lógica o la programación los cuales están ceñidos rigurosamente a reglas establecidas [6]

El lenguaje natural es enriquecido por su vocabulario y construcciones, también se establecen características como la flexibilidad, ambigüedad e indeterminación permitiendo la variedad en al interpretación dependiendo de la situación, lo cual resulta ventajoso en el momento de efectuar la comunicación humana, pero al momento de enfrentarse al procesamiento computacional dichas características se presentan como un problema ya que dificultan la aplicación de procesos de razonamiento, caracterización y formalización otros [7].

Procesamiento natural del lenguaje (PNL) es el campo de estudio que busca entender cómo funciona el lenguaje, su construcción, la generación de nuevo lenguaje, así como todas las tareas que tienen relación con el tratamiento del lenguaje. Entre estas tareas tenemos la generación de nuevo texto, traducciones de un idioma a otro, preguntas y respuestas, generar resumen, chatbots entre otros [8].

Gracias a la aparición del aprendizaje profundo en la última década del aprendizaje profundo es que se han generado avances significativos en el campo del PNL, en la siguiente sección encontraremos distintas herramientas o modelos que han sobresalido.

A. Procesamiento natural, Inteligencia Artificial y Redes Neuronales

Como una aproximación general, se define Inteligencia Artificial (IA de aquí en adelante) según [9]como "ciencia que tiene como objetivo el diseño y construcción de máquinas capaces de imitar el comportamiento inteligente de las personas" que se relaciona directamente con el PLN, cuando se busca en investigaciones o experimentos que la máquina, sea robot o celular genere oraciones y contextos en el lenguaje hablado y común.

En estos experimentos, se han considerado diferentes modelos y procedimientos para obtener resultados de buen desempeño en la predicción de la siguiente palabra, por ejemplo, o al resumen de textos, o a la generación de código [10] de programación. Uno de los algoritmos más usados en estos modelos y experimentos investigados se denomina Redes Neuronales que buscan de cierta manera representar el funcionamiento biológico de una "Neurona humana", es decir, a traveś de operaciones matemáticos pretenden procesar y compartir información entre ellas como lo harían las neuronas en el cerebro de una persona [11]. Este algoritmo tiene diferentes tipos, así podemos encontrar, por ejemplo, Redes Neuronales Convolucionales, Recurrentes. Se muestra el esquema general de una red neuronal:

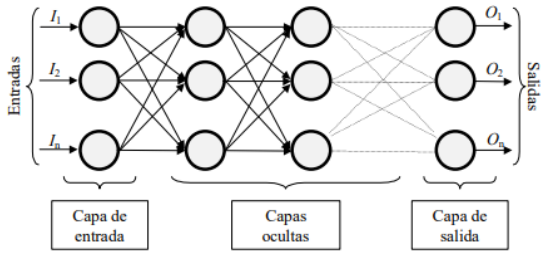

Figura 1- un esquema de una red neuronal tomado de [10]

Estos algoritmos o modelos han presentado muy buenos resultados, pero han sido diversificados con los siguientes modelos, mejorando la capacidad de respuesta a tareas de pregunta-respuesta o traducción.

B. Modelos Relevantes PLN

A continuación, se resumen los modelos que han representado un cambio significativo en los últimos años para la realización efectiva de tareas relacionadas con el PNL.

Retrocediendo en el tiempo, estos modelos inician una curva evolutiva desde el 2001, buscando un mejor comportamiento para la predicción de la siguiente palabra en una frase. Para esto "Yoshua Bengio planteó una red neuronal feed-forward", que recibe un vector de entrada con una cantidad de palabras que serían las anteriores a la que se busca predecir, con estos vectores se genera una matriz que hoy en día se conoce como word embeddings [12] [13]. En ese proceso de crecimiento y avance se planteó el uso de Redes Neuronales Recurrentes en 2010 [14]. Un poco más adelante, en 2013 se usan las Long Short-term memory (LSTM), que

“son modelos basados en RNN que añaden células de memoria para mejorar el problema del ruido cuando hay mucho contexto de por medio entre dos palabras relevantes” [12] [15]. Alrededor de esta época se desarrolla con más profundidad el concepto de Word Embeddings y para los siguientes años se vieron aplicaciones de otras Redes Neuronales, como las convolucionales en aspectos de visión por computadora y las recursivas, que “establecen relaciones jerárquicas que toman forma de árbol” [12] [16]. Finalmente llegamos al 2017, momento en el que se inicia el auge del PNL computarizado y se plantea el Transformer Language Model.

1.Transformer Language Model

Previo a este concepto, la solución que se adaptaba mejor a problemas de traducción era la computación secuencial que forma la base de la GPU Neuronal Extendida y que se refleja en los modelos de redes neuronales convolucionales, en los cuales la cantidad de operaciones va aumentando según la distancia de posiciones de las señales ingresadas y señales resultado. Este aumento en el transformer se reduce a una constante de operaciones [17].

El Transformador fue propuesto como un mecanismo de atención para conectar un codificador-decodificador de una Red Neuronal Recurrente o una Red Neuronal Convolucional, evitando las recurrencias y convoluciones buscando un mejor desempeño y rendimiento a través del mecanismo de atención [17].En resumen, el Transformer considera una arquitectura de modelo que se basa completamente en un mecanismo de atención para identificar dependencias entre la entrada y la salida. Además, este modelo se orienta muy bien a la paralelización o procesamiento simultáneo.

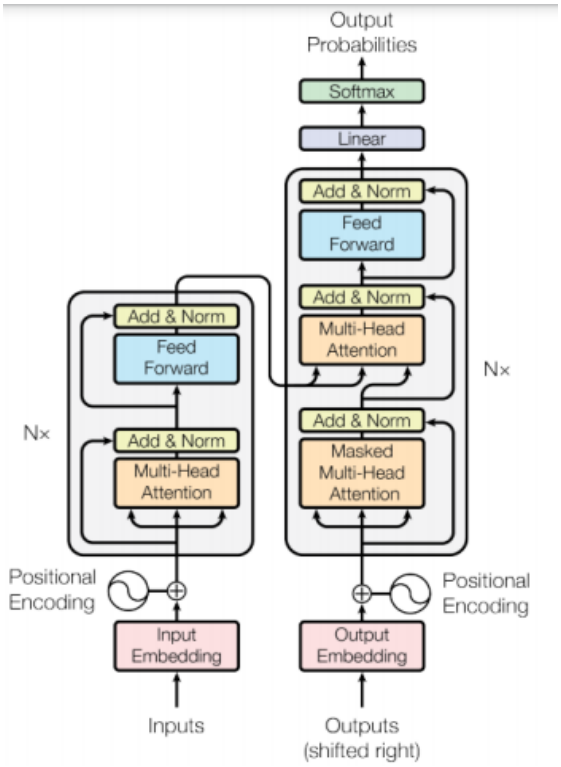

Figura 2-Arquitectura del modelo Transformador descrita en artículo “Attention Is All You Need” [17]

Este modelo permitió que BERT y GPT surgieran y se desarrollaran y mostrando resultados exitosos en diversas tareas de PNL.

2. GPT

GPT, que significa Generative Pre-Trained Transformer [18] y es en resumen un modelo de procesamiento del lenguaje generado por la compañía de investigación en el área de Inteligencia Artificial [19] (o en inglés Artificial Intelligence) OpenAI. GPT se ha considerado para extracción de relaciones de conceptos en un texto, traducciones o respuestas a preguntas. Utiliza la arquitectura basada en transformadores para consolidar Modelos del Lenguaje. Comprende el entrenamiento de un conjunto de datos en el cual, el modelo aprende las tareas como respuesta a preguntas, traducción automática o resumen de textos sin ninguna supervisión explícita [20]..

GPT-2, es una versión del modelo que salió oficialmente el 5 de noviembre de 2019, "es un transformador de parámetros de 1.5 Billones que logra resultados de vanguardia en 7 de 8 conjuntos de datos de modelado de lenguaje probados en una configuración de cero" [20].. Sus resultados han generado textos con coherencia y han permitido mejorar la tarea de respuesta de los chatbots.

GPT-3 es la tercera generación del modelo de PNL de la organización OpenIA, es autorregresivo y usa como su predecesor Aprendizaje Profundo con el fin de producir texto similar al del humano [21]. No está disponible para el público en general, su acceso se permite a una versión Beta a través de un formulario que pide toda la información necesaria para asegurar un uso ético del modelo. El acceso se permite a desarrolladores con proyectos prometedores o a científicos que están buscando la mejora constantemente del modelo [1]. Se remarca sobre la tarea específica para la que fue diseñado el sistema GPT y es la generación de texto a partir de unos datos de entrada, es decir, predecir la siguiente palabra en una secuencia de palabras. Otra característica del modelo es que no aprende, lo que hace es utilizar la información que se ingresa ya sea para responder lo que se le pregunta o para hacer la tarea que se le pida si se le induce a ello. "GPT-3 comenzó a escribir comunicados de prensa o manuales técnicos, resumir artículos, permitirte tener conversaciones con personajes históricos, generar historias cortas o canciones escritas en base al estilo de un artista específico, escribir correos electrónicos completos basados en algunos puntos dados" [22]. Los anteriores ejercicios mencionados establecen la generación de texto en LN a partir de texto en LN. Por otra parte, se han planteado otra clase de pruebas como la generación de texto en lenguaje formal a partir del LN, aplicado al lenguaje de máquina (Código de programación, datos estructurados) [22].

3. BERT

BERT significa en inglés Bidirectional Encoder Representations from Transformers y en español Representación de Codificador Bidireccional de Tranformadores que es un método de representación de lenguaje utilizado en tareas como respuesta a preguntas. Este modelo se diseñó para pre-entrenar representaciones bidireccionales profundas sin etiqueta para todas las capas [23]..

Este modelo permite, por medio de las representaciones pre-entrenadas evitar tareas de ingeniería complejas y pesada de arquitecturas de tareas específicas [23].

Para la aplicación de este modelo se consideran dos etapas fundamentales: pre-entrenamiento y ajuste. Este método fue creado y publicado en 2018 por Google y los encargados del proyecto: Jacob Devlin y sus colaboradores. Para un detalle más amplio se puede visitar el repositorio Git: https://github.com/google-research/bert.

4. Ajuste del Modelo del Lenguaje Universal

Este modelo se propuso como un método de transferencia de aprendizaje inductivo y efectivo que puede ser aplicado a cualquier tarea de Procesamiento Natural del Lenguaje (PNL o por sus siglas en inglés, natural language processing, NLP) [24]. Para la clasificación de texto, este método tuvo un desempeño considerable con pocos ejemplos etiquetados [24] .

Inicialmente este modelo se utilizó bastante y con buenos resultados en el ámbito de la visión de computadoras, pero se ajustó al PNL en la tarea de clasificación de texto para tareas de identificación de Spam, fraude y detección de chat-robots, entre otras.

Para la transferencia inductiva se usa el ajuste de incrustación de palabras pre entrenadas (fine-tuning pretrained word embeddings en inglés).

5. ELMo

ELMo (Embeddings from Language Models) "aprende incrustaciones de palabras contextuales como una función de una oración entera de entrada", permitiendo trabajar la polisemia[1] de las palabras [25].

Estos modelos entrenan 2 capas de BiLSTM [26], convolucionales que aprenden un modelo del lenguaje direccional (hacia adelante que predice la palabra contigua y hacia atrás para predecir símbolos) de un gran cuerpo de texto [25].

Aplicaciones de Procesamiento Natural del Lenguaje

Lo que más destaca en esta evolución del PNL son las aplicaciones, que han asombrado por la coherencia en la generación de los textos y en la traducción automática o en las implementaciones de chatbots, que responden de manera muy semejante a la del ser humano.

1. **Meena Chatbot de Google**

Para febrero de 2020, google y el grupo de investigadores presentaron Meena, un chatbot de dominio abierto, que involucra una red neuronal de 2.6 billones (miles de millones) de parámetros y que fue entrenada con conversaciones de las redes sociales de dominio público [27]. El objetivo del entrenamiento fue disminuir la perplejidad del siguiente token [2] [28].

Igual que otros chatbots de dominio abierto como MILABOT [29] busca ajustar la conversación para conseguir conversaciones de temas diferentes. En contraste los chatbots de dominio cerrado responden a palabras clave o tareas específicas.

Como otros chatbots presenta problemas de ambigüedades o ideas vagas y genéricas en el momento de llevar la conversación [27].

2. **Blender chatbot de Facebook**

En abril de 2020, Facebook AI hace el lanzamiento de BlenderBot, un chatBot abierto (open source) que se expresa de forma muy humana, que combina habilidades de conversación como empatía, conocimiento y personalidad, según los evaluadores finales [30].

Este chatbot utiliza técnicas de decodificación mejoradas y un modelo con 9.4 mil millones de parámetros. Al ser un recurso open source, investigadores pueden utilizarlo para mejorarlo o estudiarlo.

En la investigación de Facebook IA para construir BlenderBot se consideraron una mezcla de habilidades,

1 Fenómeno del lenguaje que consiste en que una misma palabra tiene varios significados.

2 Tokenización: Dada una secuencia de caracteres y una unidad de documento definida, la tokenización es la tarea de cortarlo en pedazos, llamados tokens, quizás al mismo tiempo desechando ciertos caracteres, como la puntuación.

más allá de un escalamiento de un modelo, buscando respuestas con buen desempeño en conversación, ajustando el perfil de máquina a respuestas de una persona. Así, se enfocó el modelo en personalidad y simpatía, conocimiento y empatía [31] tomando de base la configuración de "Combinación de Habilidades Conversacionales o Blend Skill Talks" planteado y evaluado en [32].

Algunas conclusiones en la construcción de este este chatbot es que los resultados dependen de las configuraciones que se elijan. Entonces el modelo se puede ver afectado por la elección de temas específicos en la conversación, tiempos de la conversación, longitud de frases. Los mejores resultados se presentaron cuando se construyeron conversaciones cortas y en turnos variables. Estos resultados superaron a Meena en evaluaciones humanas de simpatía.

A pesar del buen desempeño, el chatbot muestra inconvenientes cuando se le pregunta lo suficiente. Además, presenta una tendencia a mantener lenguaje simple y a repetir frases de uso frecuente. Se plantea, al final del estudio, algunas formas para mejorar estos problemas como entrenamiento de probabilidad, mecanismos de recuperación y refinamientos.

3. Traducción automática

Uno de los campos de mayor aplicación de PNL es la traducción automática. Por ejemplo, se han estudiado y adaptado modelos como BERT con traducción automática neuronal (en inglés Neural Machine Translation).

En la adaptación de BERT [33], se utilizó éste como incrustación contextual [34], lo que en primera medida extrae representaciones para una secuencia de entrada, y luego las representaciones se fusionan con cada capa del codificador y decodificador del modelo NMT a través de mecanismos de atención Finalmente hacen experimentos de traducción para oraciones y documentos utilizando técnicas supervisadas y no supervisadas, obteniendo muy buenos resultados. El código de este proyecto se puede encontrar en https://github.com/bert-nmt/bert-nmt. En las conclusiones de esta investigación, se plantea la aceleración del proceso de inferencia, la aplicación de la fusión construida a pregunta-respuesta y la búsqueda de una compresión del consolidado BERT-NMT.

Otra adaptación de BERT, propuso un enfoque de entrenamiento concertado para aprovechar BERT en NML [35].

4. Resumen de texto

Antes del auge de los Transformadores hacia 2017, la producción de LN utilizaba en mayor medida modelos estadísticos, formalizados y con reglas heurísticas[3]. Sus resultados definitivamente eran muy pobres combinando palabras y frases de temas que no tenían ninguna relación cercana (poemas y ensaladas) generando información incoherente. A partir de los transformadores y GPT-3 que por exposición repetida logra ajustar bastante bien las frases, con el contexto generado puede producir muy buenos resúmenes o extracciones de texto [36] [37].

Se asume el concepto de resumen de un texto según [37], que lo expone como " el proceso de reducir el tamaño de un documento mientras se conserva el significado)". Existen dos técnicas, a nivel general para resumir texto: técnica extractiva, que considera las oraciones tal y como aparecen en el texto original y técnica abstractiva, que genera un texto nuevo para generar el resumen [37].

3 Técnica de la indagación y del descubrimiento.

Una de las investigaciones que compara seis modelos diferentes (dos con arquitectura RNR, otro con RNC, otros dos Transformer y otro combinado de transformer y aprendizaje reforzado) teniendo en cuenta artículos científicos, obtuvo que "aquellos modelos que utilizan un decodificador jerárquico para modelar la estructura del documento presentan un mejor desempeño que el resto. Uno de los modelos de Transformador utilizado en la investigación se denomina ProphetNet y es abstractivo, realizando un paso extractivo antes de generar el resumen. Otro de los modelos transformer se llama único lenguaje Transformador y es similar a GPT, que antes de generar el resumen, hace un paso extractivo para condicionar el modelo del lenguaje Transformador con información relevante del documento.

5. Generación de documentos o literatura

En los campos de la literatura, ya se habla de la generación de textos literarios, es así como Yi Liao y colegas presentan un método para generar poesía china clásica y acróstica, incluso con alta calidad utilizando GPT [38], formando así un sistema de generación de poesía. Para pruebas y evaluaciones se ha lanzado un mini programa de demostración en línea en Wechat.

Otra iniciativa en donde se encuentran diversidad de poemas, en inglés, generados con GPT-3 es el proyecto gwern: https://www.gwern.net/GPT-3#miscellaneous-poetry.

En otro proyecto de investigación se utilizan y comparan BERT y GPT-2 para la generación de texto obteniendo que los dos modelos sacan ventaja de las capas de atención consiguiendo muy bien rendimiento en puntos de referencia de modelado de idiomas. Se identifica que los resultados dependen de qué tanto el modelo esté relacionado o entrenado con información de determinado tema, además GPT-2 no puede generar texto a partir de una palabra poco corta. En cuanto a BERT se encuentra que presenta problemas en los homónimos o sinónimos [39]..

C. Ingeniería de software (ciclo de vida del software)

La ingeniería de software en sus principios, se establece como el campo de tecnología encargada del diseño y construcción sistemática de producto de software, En una definición más puntual encontramos la dada por la IEEE en 1990 que la define como "la aplicación de un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento del software" [40].

A lo largo de la evolución de la ingeniería de software se han planteado distintos modelos de proceso (Cascada, evolutivo, lineal, otros) que enmarcan una serie actividades, acciones o tareas así como, su relación y dependencia con el fin de generar de la mejor manera posible productos de software este proceso se denomina ciclo de vida del software [41]. Indistintamente del modelo de proceso o metodología de desarrollo que se escoja y de su flujo de aplicación, se encuentran unas actividades generales que se encuentran presentes en el ciclo de vida de los diferentes modelos.

Especificación de requerimientos: Es el proceso de descubrir el propósito del software identificando las funcionalidades requeridas a través de los interesados y sus necesidades, así como identificar las restricciones operacionales [42].

Diseño e Implementación: Se propone un diseño según los requerimientos establecidos y luego se realiza construcción del software de acuerdo al diseño aprobado.

Validación: El software debe ser validado, con el fin de asegurar el cumplimiento de lo requerido por el cliente [43]

Mantenimiento: El software requiere evolucionar, para suplir las necesidades cambiantes del cliente que se reflejarán en nuevos requisitos [44].

1. Lenguaje natural e ingeniería de software

Como hemos podido ver en las secciones anteriores el LN y su estudios han presentado múltiples herramientas para abordar diferentes problemas o necesidades que se presenta en cuanto a comunicación, en esta sección se establecerá las necesidades o dificultades que se presentan en la distintas tareas que hacen parte del ciclo de vida de software, así como estableciendo la relación de ellas con el LN, permitiéndonos vislumbrar la posibilidad acotar soluciones lo suficientemente amplia como para que la generación de productos de software se convierta en una proceso realmente agil.

Comenzando según el flujo de vida del software del desarrollo de software se encuentra que la especificación o recolección de requerimientos se realizan principalmente a través de LN facilitando la interacción entre todos los involucrados, aunque
genera falencias en cuanto a la ambigüedades y falta de completitud de los requerido [45]. En el marco de de la recolección de requisitos también encontramos inconsistencias las cuales se establecen cuando un requisito es opuesto a otro o dentro de su redacción hay contradicciones, también existe la duplicidad al encontrar que dos requerimientos significan los mismo pero con diferentes palabras [46]. También tenemos que este tipo de dificultades afectan la estimación de tiempos de desarrollo, a pesar que existen múltiples metodologías (Juicio de Expertos , Descomposición de tareas, Estimación por analogías, Método Paramétrico, Puntos de Casos de U, otros) para resolver este inconveniente en general el desfase de la estimación del esfuerzo requerido en tiempo suele ser notorio esto se da usualmente, esto se da generalmente por falta de información, poca experiencia, cambios no concertados y las dificultades en la especificación de requisitos anteriormente mencionadas [47].

En las etapa de diseño e implementación los requerimientos y la comunicación son indispensables ya que de la comprensión y entendimiento de ellos se generan los productos que a su vez son insumos para las mismas u otras etapas. Al hablar de la generación de prototipos o diseños gráficos para productos de software, los resultados suelen ser devueltos en distintas ocasiones como consecuencia de una interpretación errónea de lo solicitado [48].

2. GPT-3 aplicaciones reales a la ingeniería de software

Se resalta la relevancia que a tomado GPT en los últimos años y sobre todo con la publicación de su modelo más reciente GPT-3, como se mencionaba anteriormente GPT es un modelo entrenado para generar texto a partir de texto, esta tarea puede ser enfocada de distintas maneras donde una de ellas es la generación de código de programación o estructuras de texto (datos) a partir del LN [1] es aquí donde se establece la relación de este modelo directamente con la Ingeniería de Software. Desde que salió el modelo GPT-3 en junio del 2020, los usuarios que han podido acceder al modelo de uso restringido han generado una gran cantidad herramientas con base a la APl que suponen una gran ayuda para los ingenieros una vez se perfeccionen. A continuación, revisaremos algunos ejemplos que involucran actividades propias del ciclo de vida del software.

Búsqueda semántica para establecer la dificultad de una tarea, que puede ser extrapolado a la generación de de estimación para puntos de historias de usuarios.

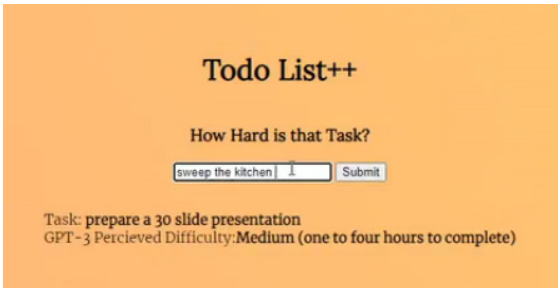

Figura 3. Ejemplo de establecer la dificultad de una tarea [49].

Generación de diseños a través de instrucciones básicas en inglés, al interactuar con el API se establece que a una entrada descriptiva del producto de diseño que quiere obtener, presenta una estructura interpretable por la herramienta Sigma que es una herramienta de diseño que permite representar los prototipos diseñados de interfaz de usuario [50]:

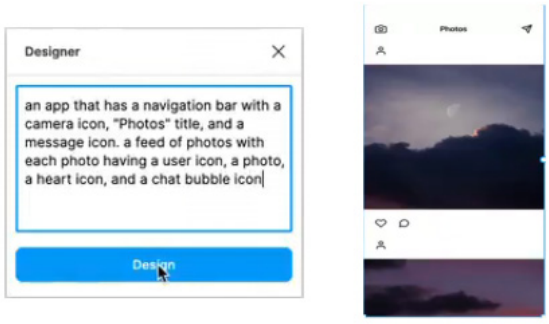

Figura 4. Conversión de LN a interfaz de usuario básica [49].

En el marco de desarrollo de software se encontraron generadores de código a partir de instrucciones en lenguaje natural como para consultas SQL, interfaz en de Javascript (Three.js, React app).

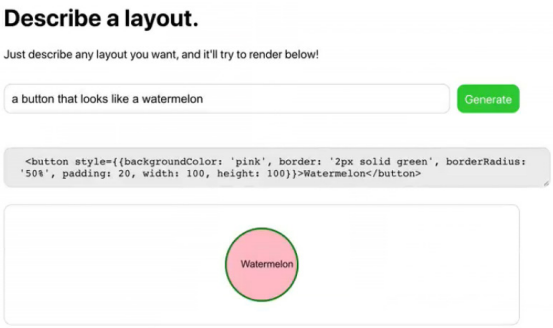

Figura 5. A Partir de LN genera diseño en código para web [49].

Figura 6. a partir de LN genera código ejecutable de la librería Three.jsx [49].

También se pudieron ver ejemplos de generación de código orientado a Devops en AWS y comandos linux. Como herramientas para apoyar el desarrollo también se encontró una que permite a través de la introducción de un fragmento de código y una pregunta relacionada con la funcionalidad, la herramienta responde con la explicación de lo solicitado.

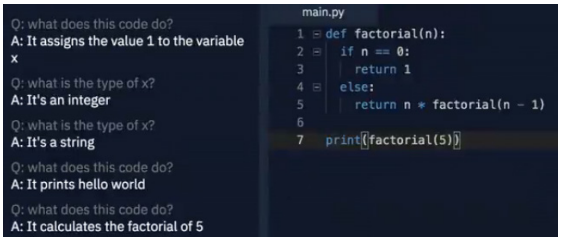

Figura 7. Presenta respuesta respecto a un código dado.

También se pudieron ver ejemplos de generación de código orientado a Devops para Amazon Web Services (AWS) y otro de comandos linux.

Figura 8. A partir de texto en LN se generan comandos para AWS.

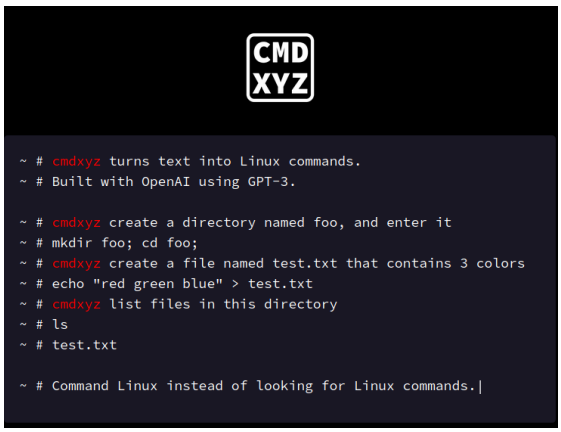

Figura 9. A partir de texto en LN generar comandos en linux

III. Conclusiones

La generación de productos de software está mediados por el lenguaje natural en la comunicación, a través de sus distintas etapas permitiendo de esta manera que los resultado finales sean dependientes de la compresión y precisión en dicha comunicación.

El procesamiento natural del lenguaje y sobretodo los modelos GPT han abierto notablemente la brecha de comunicación e interacción que tienen el ser humano y la máquina, permitiendo de cierto modo democratizar la tecnología.

Es posible que GPT-3 haya establecido el punto de partida para que la interacción entre hombre y máquina cambie, permitiendo la generalización de software de una manera mucho más rápida y directa, proporcionado

al usuario la posibilidad de representar sus necesidades de un producto de software a partir del lenguaje natural.

A pesar que GPT-3 se muestra prometedor la limitación de uso restringe la posibilidad de generar aplicaciones centradas en las necesidades generales de la comunidad vinculada al desarrollo de software.

Recomendaciones

Si fuera posible más adelante a través de instituciones o centros educativos poder acceder a GPT-3 sería interesante ponerlo en práctica en las materias relacionadas con el desarrollo de software, con el fin de evaluar que otras soluciones se pueden plantear con base en este modelo como usar el lenguaje natural como interfaz de usuario para interactuar con las aplicaciones.

Referencias

[1] OpenAI, «Blog OpenAI,» OpenAI, 22 09 2020. [En línea]. Disponible: https://openai.com/blog/openai-licenses-gpt-3-technology-to-microsoft/. [Último acceso: 12 11 2020].

[2] OpenAI, «Api Beta GPT-3,» OpenAI, 22 06 2020. [En línea]. Disponible: https://beta.openai.com/. [Último acceso: 12 11 2020].

[3] T. B. Brown y e. al., «Language Models are Few-Shot Learners,» arXiv, pp. 1-75, 2020.

[4] Z. Gao, a. A. Feng, a. X. Song y X. Wu, «Target-Dependent Sentiment Classification With BERT,» IEEE Access, vol. 7, pp. 154290-154299, 2019.

[5] B. Manrique-Losada, «PROCESAMIENTO DE LENGUAJE NATURAL PARA ADQUISICIÓN DE CONOCIMIENTO: APROXIMACIONES DESDE LA INGENIERÍA DE REQUISITOS,» Revista QUID, nº 24, pp. 69-78, 2015.

[6] A. Cortez Vásquez, H. Vega Huerta y J. Pariona Quispe, «Procesamiento de lenguaje natural,» Revista de Ingeniería de Sistemas e Informática, vol. 6, nº 2, pp. 45-54, 2009.

[7] M. Vicente, C. Barros, F. S. Peregrino, F. Agullo y E. Lloret, «La generación de lenguaje natural: análisis del estado actual,» Computación y Sistemas, vol. 19, nº 4, pp. 721-756, 2015.

[8] I. GIL LEIVA y J. V. RODRÍGUEZ MUÑOZ, «EL PROCESAMIENTO DEL LENGUAJE NATURAL APLICADO AL ANÁLISIS DEL CONTENIDO DE LOS DOCUMENTOS,» Revista General de Información y Documentación, vol. 6, nº 2, pp. 205-218, 1996.

[9] L. Munárriz, Fundamentos de inteligencia artificial, Murcia: Universidad de Murcia, 1994, pp. 19-21.

[10] P. Yalla y N. Sharma, «Integrating Natural Language Processing and Software Engineering,» International Journal of Software Engineering and Its Applications, vol. 9, nº 11, pp. 127-136, 2015.

[11] D. J. Matich, «Redes Neuronales: Conceptos Básicos y Aplicaciones.,» 03 03 2001. [En línea]. Disponible: https://www.frro.utn.edu.ar/repositorio/catedras/quimica/5_anio/orientadora1/monograias/matich-redesneuronales.pdf. [Último acceso: 5 11 2020].

[12] J. A. Moncho, «Modelos contextuales e incontextuales de palabras para Twitter-Trabajo de Grado,» Universidad Politécnica de Valencia, 20 08 2019. [En línea]. Disponible: https://riunet.upv.es/bitstream/handle/10251/151024/Arias%20-%20Modelos%20contextuales%20e%20incontextuales%20de%20palabras%20para%20Twitter.pdf?sequence=1&isAllowed=y. [Último acceso: 15 11 2020].

[13] Y. Bengio, R. Ducharme, P. Vincent y C. Jauvin, «A Neural Probabilistic Language Model,» Journal of Machine Learning Research, nº 3, p. 1137–1155, 2003.

[14] T. Mikolov, M. Karafiat, L. Burget, J. “. Cernock y S. Khudanpur, «Recurrent neural network based language model,» INTERSPEECH 2010, vol. 1, nº 1, pp. 1045-1048, 2010.

[15] A. Graves, «Generating Sequences With Recurrent Neural Networks,» Neural and Evolutionary Computing, pp. 1-43, 06 2014.

[16] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng y C. Potts, Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, Seattle, Washington, USA: Association for Computational Linguistics, 2013, pp. 1631--1642.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser y I. Polosukhin, «Attention Is All You Need,» de 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, 2017.

[18] C. Alt, M. Hubner y L. Hennig, «Fine-tuning Pre-Trained Transformer Language Models to Distantly Supervised Relation Extraction,» arXiv, p. 11, 2019.

[19] https://openai.com/about/, «OpenIA,» OpenIA, 05 11 2015. [En línea]. Disponible: https://openai.com/about/. [Último acceso: 20 11 2020].

[20] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei y I. Sutskever, «Language Models are Unsupervised Multitask Learners,» OpenAI, pp. 1-24, 2018.

[21] L. Floridi y M. Chiriatti, «GPT-3: Its Nature, Scope, Limits, and Consequences,» Minds and Machines, pp. 1-14, 2020.

[22] P. Perazzo, «GPT-3 and the Rise of Human-centric Adaptive Software - Intro,» Perazzo, P., 29 08 2020. [En línea]. Disponible: Ppaolo.substack.com. [Último acceso: 3 11 2020].

[23] J. Devlin, M.-W. Chang, K. Lee y K. Toutanova, «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,» arXiv, pp. 1-16, 2019.

[24] J. Howard y S. Ruder, «Universal Language Model Fine-tuning for Text Classification,» arXiv, pp. 1-12, 2018.

[25] J. Gomez-Perez, R. Denaux y A. Garcia-Silva, A Practical Guide to Hybrid Natural Language Processing: Combining Neural Models and Knowledge Graphs for NLP, Suiza: Springer International Publishing, 2020.

[26] M. Rhanoui, M. Mikram, S. Yousfi y S. Barzali, «A CNN-BiLSTM Model for Document-Level Sentiment Analysis,» Machine Learning and Knowledge Extraction, vol. 1, pp. 832-847, 2019.

[27] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu y Q. V. Le, «Towards a Human-like Open-Domain Chatbot,» arXiv, vol. 1, pp. 1-38, 2020.

[28] «The Stanford Natural Language Processing Group,» The Stanford University, 07 04 2009. [En línea]. Disponible: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html. [Último acceso: 19 11 2020].

[29] Serban, I. V., C. Sankar, M. Germain, S. Zhang, Z. Lin, S. Subramanian, T. Kim, M. Pieper, S. Chandar, N. Rosemary Ke, S. Rajeshwar, A. de Brebisson, J. M. Sotelo, D. Suhubdy y V, «A Deep Reinforcement Learning Chatbot,» arXiv, pp. 1-40, 2017.

[30] F. AI, «Facebook AI,» Facebook, 20 04 2020. [En línea]. Disponible: https://ai.facebook.com/blog/state-of-the-art-open-source-chatbot/. [Último acceso: 5 11 2020].

[31] S. Roller, E. Dinan, N. Goyal, D. Ju, M. Williamson, Y. Liu, J. Xu, M. Ott, K. Shuster, E. M. Smith, Y.-L. Boureau y J. Weston, «Recipes for building an open-domain chatbot,» arXiv, pp. 1-25, 2020.

[32] E. M. Smith, M. Williamson, K. Shuster, J. Weston y Y.-L. Boureau, «Can You Put it All Together: Evaluating Conversational Agents' Ability to Blend Skills,» arXiv, pp. 1-10, 2020.

[33] J. Zhu, Y. Xia, L. Wu, D. He, T. Qin, W. Zhou, H. Li y T.-Y. Liu, «INCORPORATING BERT INTO NEURAL MACHINE TRANSLATION,» arXiv, pp. 1-18, 2020.

[34] Q. Liu, M. J. Kusner y P. Blunsom, «A Survey on Contextual Embeddings,» arXiv, pp. 1-13, 2020.

[35] J. Yang, M. Wang, H. Zhou, C. Zhao, W. Zhang, Y. Yu y L. Li, «Towards Making the Most of BERT in Neural Machine Translation,» Proceedings of the AAAI Conference on Artificial Intelligence, vol. 41, nº 5, pp. 9379-9385, 2020.

[36] K. Elkins y J. Chun, «Can GPT-3 Pass a Writer's Turing Test?,» Journal of Cultural Analytics CA, vol. 1, pp. 1-16, 2020.

[37] R. B. Soler y J. Kalita, «Abstractive and mixed summarization for long-single documents,» arXiv, pp. 1-9, 2020.

[38] Y. Liao, Y. Wang, Q. Liu y X. Jiang, «GPT-based Generation for Classical Chinese Poetry,» arXiv, pp. 1-16, 2019.

[39] D. Muñoz Montesinos, «MODERN METHODS OF TEXT GENERATION,» arXiv, p. 22, 2020.

[40] G. PANTALEO y L. RINAUDO, Ingeniería de Software, Buenos Aires, Argentina: Alfaomega Grupo Editor, 2015.

[41] I. SOMMERVILLE, Ingeniería de Software, Reino Unido: Pearson educación, 2005.

[42] R. Gacitúa Bustos, «Procesamiento de Lenguaje Natural en Ingeniería de Requisitos: Contribuciones Potenciales y Desafíos de Investigación,» 6 05 2015. [En línea]. Disponible: https://eventos.spc.org.pe/cibse2015/pdfs/2_Tutorial15.pdf. [Último acceso: 3 11 2020].

[43] R. S. Pressman, Ingeniería de software, un enfoque práctico, McGraw-Hill, 2006.

[44] Universidad de Sevilla, «PROBLEMÁTICA DE LOS PROYECTOS SOFTWARE,» [En línea]. Disponible: http://bibing.us.es/proyectos/abreproy/70193/fichero/5.+PROBLEM%C3%81TICA+DE+LOS+PROYECTOS+SOFTWARE.pdf. [Último acceso: 4 11 2020].

[45] G. H. a. J. D. C. Litvak, «C. Litvak, G. Hadad and J. Doorn, “Procesamiento de Lenguaje Natural para Estudiar Completitud de Requisitos”, UNLaM, pp. 498-502, 2015.,» UNLaM, pp. 498-502, 2015.

[46] C. V. C. a. S. G. A. J. Guzmán Luna, «“Un modelo de procesamiento de lenguaje natural para la detección de errores en requisitos de software,» evista Virtual Universidad Católica del Norte, vol. 46, pp. 69-186, 2015.

[47] W. R. B. R. Ramírez Blauvelt, «Metodología para la estimación de proyectos de desarrollo de software para la empresa sophos banking solutions s.a.s.,» Maestría en administración, Universidad EAFIT, 2015.

[48] J. L. Duarte, Metodología para la detección de requerimientos subjetivos en el diseño de producto-Doctorado, Cataluña: Universitat Politècnica de Catalunya, 2006.

[49] A. Joshi, «GPT3 Examples,» GPT3 Examples, 2 08 2020. [En línea]. Disponible: https://gpt3examples.com/. [Último acceso: 20 11 2020].

[50] J. Singer, «GPT-3 (what, why, how), Figma Disponible for you,» GPT3 Examples, 12 05 2020. [En línea]. Disponible: https://ibuildmyideas.substack.com/p/i-build-my-ideas-8-071920. [Último acceso: 04 11 2020].