

Consultas SparQL a estructuras RDF

BRYAN ASMAT, DANILO BLAS, VICTOR CHAVEZ, WILLIAM GUADALUPE

Universidad Nacional de Ingenieria

Lima, Perú

basmatf@uni.pe, iblass@uni.pe, vchavezb@uni.pe, wguadalupeq@uni.pe

Resumen—La información en la web se puede representar bajo un formato de datos llamado RDF

En el presente trabajo se implementaron consultas de tipos SPARQL que permiten hacer la búsqueda en distintas fuentes de datos.

Índice de Términos— RNN, modelos seq2seq, mecanismos de atención, transformers.

I. INTRODUCCIÓN

Una tendencia dada en el ICRL¹ y en el NAACL- HLT² celebrados en el 2019, muestra que las RNN presentan un declive en el número de publicaciones realizadas. Dicha tendencia era de esperarse, aunque las RNN son intuitivas para los datos secuenciales tienen como principal inconveniente la paralelización en su procesamiento, por lo tanto, no pueden aprovechar el factor más importante que ha impulsado el progreso en la investigación desde 2012: la potencia de cálculo. Los RNN nunca han sido populares en visión computacional y aprendizaje por refuerzo y para NLP, están siendo reemplazados por arquitecturas basadas en la atención.

Los Transformers, presentados en 2017, introdujeron un nuevo enfoque: módulos de atención. Este modelo ha conllevado a una enorme cantidad de variantes que mejoran el desempeño de tareas del NLP, como el Transformer Universal, BERT (google) o el Transformer-XL; la mayoría de modelos de vanguardia que requieren gran cantidad de datos de entrenamiento y días de entrenamiento en hardware haciéndolo costoso. Sin embargo, con el lanzamiento de librerías especializadas en NLP y arquitecturas Transformer, ahora pueden ser utilizados en datos reducidos.

De esta manera, el presente trabajo busca explorar la aplicación de estos modelos sobre el conjunto de datos reducido AG News y Multi30k. En primer lugar se realizará la definición del modelo Transformer. A continuación, se describirá el estado del arte de modelos basados en el Transformer, para diferentes tareas del NLP. Posteriormente, se explica la metodología a utilizar en los experimentos con diferentes arquitecturas. Las secciones siguientes tratarán los resultados experimentales, las conclusiones y discusiones, y los trabajos futuros que

se pueden realizar con esta nueva familia de técnicas del NLP.

II. ESTRUCTURA RDF

El modelo Transformer, propuesto en el artículo [Attention Is All You Need \[1\]](#), se basa en la auto-atención sin el uso de RNN. Como resultado, es altamente paralelo y requiere menos tiempo para entrenamiento, al tiempo que establece resultados de vanguardia en modelamiento de lenguaje y la traducción automática.

El Transformer se basa en una estructura encoder-decoder. La diferencia entre este y cualquier otro modelo es que el Transformer se basa completamente en mecanismos de atención.

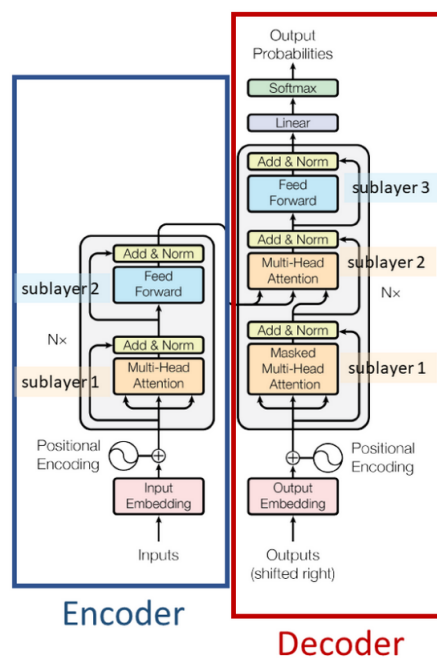


Figura 1. Encoder-Decoder del Transformer. Fuente The Transformer: Attention Is All You Need. [1]

El encoder está formado por una pila de $N = 6$ capas, cada una de las cuales está compuesta por dos subcapas: un mecanismo de atención de múltiples cabeceras y una red de alimentación directa completamente conectada más conexiones residuales [2] en ambas etapas, seguidas

¹International Conference on Learning Representations

²Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

de un procedimiento de normalización de capas [?]. El decoder se define de manera similar, solo que cada capa está compuesta de 3 subcapas: atención de múltiples cabeceras, capas completamente conectadas y atención de múltiples cabeceras enmascarada.

Este modelo fue entrenado durante 300000 pasos, aproximadamente 3,5 días, utilizando 8 GPU NVIDIA P100. De esta manera, el modelo Transformer alcanzó un puntaje BLUE [?] de 26,4 cuando se aplicó sobre el conjunto de datos *newstest2013* como un conjunto de prueba, que estableció un nuevo resultado de vanguardia.

El Transformer es una mejora con respecto a los modelos seq2seq basados en RNN, pero tiene algunas limitaciones.

- La atención solo puede ocuparse de cadenas de texto de longitud fija, lo cual implica que el texto debe dividirse en un cierto número de segmentos antes de ser alimentado al sistema como entrada.
- Esta segmentación del texto provoca la fragmentación del contexto ³. Por ejemplo, si una oración se divide en dos, se pierde una cantidad significativa de contexto. En otras palabras, el texto se divide sin considerar la oración o cualquier otro límite semántico.

III. CONSULTAS SPARQL

SPARQL, acrónimo de *SPARQL Protocol and RDF Query Language*, es un lenguaje estandarizado para hacer consultas en estructuras RDF. Es una tecnología clave en el desarrollo de la web semántica que se constituyó como recomendación oficial del W3C el 15 de enero de 2008, siendo actualizado a la versión 1.1 en 2013 [1]. En un principio SPARQL únicamente incorpora funciones para la recuperación sentencias RDF. Sin embargo, algunas propuestas también incluyen operaciones para el mantenimiento (creación, modificación y borrado) de datos.

III-A. Especificaciones

SPARQL 1.0:

- SPARQL como lenguaje de consultas para RDF, cubre la sintaxis de las consultas (QL).
- SPARQL como protocolo para RDF especifica cómo un programa debe pasar consultas SPARQL a un servicio de procesamiento de consultas SPARQL y cómo este servicio las retorna.
- Formato XML de resultados de consultas SPARQL, describe un formato simple XML para que los procesadores de consultas lo utilicen al devolver resultados.

Sparql 1.1:

- La especificación de consulta federada describe como una simple consulta puede traer data de multiples fuentes, esto hace más fácil construir aplicaciones que toman ventaja de entornos distribuidos.
- La especificación de actualización es la diferencia clave entre SPARQL 1.0 y 1.1 porque lleva a SPARQL de ser un lenguaje de consulta a algo que puede agregar datos a un conjunto de datos, reemplazarlos y eliminarlos también.
- La especificación de Descripción del servicio describe cómo un programa cliente puede preguntarle a un motor SPARQL exactamente qué características admite.
- La especificación del formato JSON de los resultados de la consulta describe un equivalente JSON del formato XML de los resultados de la consulta.
- La especificación de los formatos CSV y TSV de los resultados de la consulta describe los equivalentes de valores separados por comas y tabuladores del formato XML de los resultados de la consulta.
- La especificación del Protocolo HTTP de Graph Store amplía el Protocolo SPARQL con una API similar a REST para la comunicación entre un cliente y un procesador SPARQL sobre gráficos o conjuntos de tripletas.

III-B. Estructura RDF a utilizar

De acuerdo al artículo, las principales diferencias entre el Transformer y el Transformer Universal consisten en que el Transformer Universal aplica el encoder para un número variable de pasos para cada token de entrada/salida (T pasos), mientras que el Transformador aplica exactamente 6 capas de encoder/decoder, respectivamente. Así mismo, el Transformer Universal utiliza una representación de entrada ligeramente diferente: incluye un embedding de paso de tiempo además de la codificación posicional.

Los números variables de pasos presentes en el Transformer Universal se logran mediante el uso de **Adaptive Computation Time**, un mecanismo propuesto por Alex Graves [?] que permite la aplicación del encoder y decoder un número variable de veces.

III-C. Transformer-XL

El transformer-XL es uno de los primeros modelos exitosos en abordar el problema de poseer una longitud fija en la secuencia de entrada. En el artículo **Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context** [?] se propone este método novedoso para el modelado de lenguaje, que permite que una arquitectura transformer aprenda dependencia a largo plazo, a través de un mecanismo de recurrencia, más allá de una longitud fija sin alterar la coherencia temporal.

³Context fragmentation

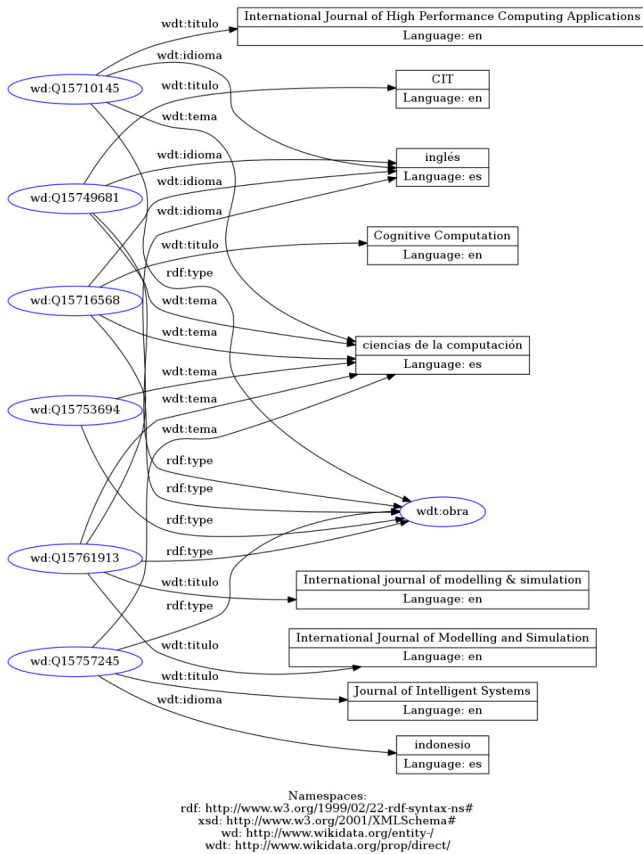


Figura 2. Transformer XL: EGráfico generado, con 30 tripletes de la estructura RDF utilizada [2]

Se introduce un mecanismo recurrente a nivel de segmento que permite que el modelo reutilice estados ocultos anteriores en el momento del entrenamiento, abordando tanto los problemas del contexto de longitud fija como la fragmentación del contexto. En otras palabras, la información histórica se puede reutilizar y se puede extender tanto como lo permita la memoria de la GPU.

Para reutilizar adecuadamente los estados ocultos, los autores proponen un mecanismo llamado codificaciones posicionales relativas que ayuda a evitar la confusión temporal. Los modelos actuales no pueden distinguir la diferencia posicional entre entradas en diferentes segmentos en diferentes capas. La codificación de posición relativa soluciona este problema al codificar el sesgo de información posicional en los estados ocultos, que difiere de otros enfoques que realizan esto como el nivel de entrada.

El transformer-XL reduce el puntaje de perplexidad SoTA anterior en varios conjuntos de datos como text8, enwiki8, One Billion Word y WikiText-103. Los autores afirman que el método es más flexible, más rápido durante la evaluación (aceleración de 1874 veces), se generaliza bien en conjuntos de datos pequeños y es

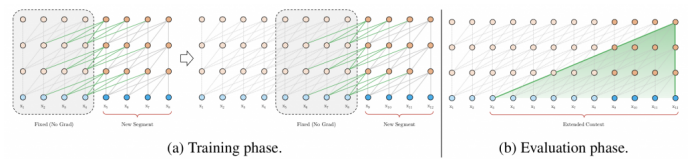


Figura 3. Transformer XL: Entrenamiento y Evaluación. Fuente: Transformer XL [?]

eficaz para modelar secuencias cortas y largas.

Así mismo, los autores proponen una nueva métrica llamada Relative Effective Context Length que proporciona una manera justa de comparar modelos que se prueban con mayores longitudes de contexto.

III-D. XLNet

XLNet es un modelo presentado en el artículo [XLNet: Generalized Autoregressive Pretraining for Language Understanding](#) [?] que es un lenguaje autoregresivo que genera la probabilidad conjunta de una secuencia de tokens basada en la arquitectura del Transformer con recurrencia. Este modelo introduce una variante de modelado de lenguaje llamada modelado de lenguaje de permutación. Los modelos de lenguaje de permutación están entrenados para predecir un token dado el contexto anterior como el modelo de lenguaje tradicional, pero en lugar de predecir los tokens en orden secuencial, predice los tokens en un orden aleatorio.

Además de usar el modelado de lenguaje de permutación, XLNet mejora BERT al usar el Transformer XL como su arquitectura base. XLNet utiliza las dos ideas clave de Transformer XL: embeddings posicionales relativas y el mecanismo de recurrencia. Los estados ocultos del segmento anterior se almacenan en caché y se congelan mientras se realiza el modelado del lenguaje de permutación para el segmento actual. Como todas las palabras del segmento anterior se usan como entrada, no es necesario conocer el orden de permutación del segmento anterior.

III-E. BERT

Este modelo [?], publicado el 2018, está conformado por una pila de bloques Transformers, la cual está pre-entrenada en un corpus de dominio general con 800 millones de palabras. Este modelo obtiene un mejor rendimiento que modelos anteriores a su publicación debido a su naturaleza bidireccional donde la atención se centra en toda la secuencia.

III-F. RoBERTa

Es un modelo introducido en Facebook. El enfoque BERT robustamente optimizado RoBERTa [?], es un re-entrenamiento de BERT con una metodología de entrenamiento mejorada, un 1000 % más de datos y potencia de cálculo. Para mejorar el procedimiento de entrenamiento,

RoBERTa elimina tareas como Next Sentence Prediction (NSP) del entrenamiento previo de BERT e introduce el enmascaramiento dinámico para que el token enmascarado cambie durante las épocas de entrenamiento.

RoBERTa supera a BERT y XLNet en los resultados de referencia de [GLUE](#).

III-G. DistilBERT

Este modelo aprende una versión destilada (aproximada) de BERT, que retiene el 95 % de rendimiento pero utiliza solo la mitad del número de parámetros. Específicamente, no tiene embeddings de tipo token, pooler y retiene solo la mitad de las capas del BERT de Google.

DistilBERT [?], utiliza una técnica llamada distillation, que se aproxima a BERT de Google. La idea es que una vez que se ha entrenado una gran red neuronal, sus distribuciones de salidas completas se pueden aproximar usando una red más pequeña. Una de las funciones clave de optimización utilizadas para la aproximación posterior en las estadísticas bayesianas es la divergencia de Kulback-Leiber.

IV. CONCLUSIONES

- Se ha realizado clasificación multiclase utilizando la librería `simpletransformers` y modelos pre-entrenados, basados en BERT, obteniendo buenos resultados en métricas de clasificación, especialmente del modelo distilBERT, que tiene un menor número de parámetros que BERT y que utiliza el aprendizaje `teacher-students`, donde se entrena a una red de `students` para imitar la distribución de salida completa de la red del `teacher` (su conocimiento) y retiene el rendimiento de BERT.
- En la tarea de traducción, se realizaron comparaciones entre distintos modelos `seq2seq` y los modelos `transformers`, donde estos últimos obtuvieron mejores resultados. Sin embargo el modelo Transformer XL modificado no logró superar al Transformer original.

REFERENCIAS

- [1] <https://www.w3.org/TR/rdf-sparql-query/>.
- [2] Página web para generar grafos a partir de una estructura RDF: .
- [3] Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton. Layer Normalization (2016). Disponible en <https://arxiv.org/abs/1607.06450>.
- [4] Kishore Papineni, Salim Roukos, Todd Ward, Wei-jing Zhu (2002). BLEU: a Method for Automatic Evaluation of Machine Translation, pp. 311-318. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL).
- [5] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, Łukasz Kaiser. Universal Transformers (2018). ICLR 2019. Disponible en <https://arxiv.org/abs/1807.03819>.
- [6] Alex Graves. Adaptive Computation Time for Recurrent Neural Networks (2017). Disponible en <https://arxiv.org/abs/1603.08983>.
- [7] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context (2019). Disponible en <https://arxiv.org/abs/1901.02860>.

- [8] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding (2019). Disponible en <https://arxiv.org/abs/1906.08237>.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (2018). Disponible en <https://arxiv.org/abs/1810.04805>.
- [10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach (2019). Disponible en <https://arxiv.org/abs/1907.11692>.
- [11] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter (2019). Disponible en <https://arxiv.org/abs/1910.01108>.
- [12] Pytorch-Transform: Implementación de transformer para NLP. https://pytorch.org/hub/huggingface_pytorch-transformers/.
- [13] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Jamie Brew. Transformers: State-of-the-art Natural Language Processing (2019), HuggingFace Inc. Disponible en <https://arxiv.org/abs/1910.03771>.
- [14] SimpleTransformers <https://pypi.org/project/simpletransformers/>.
- [15] Xiang Zhang, Junbo Zhao, Yann LeCun. Character-level Convolutional Networks for Text Classification (2015). Disponible en <https://arxiv.org/abs/1509.01626>.
- [16] Desmond Elliott, Stella Frank, Khalil Sima'an, Lucia Specia. Multi30K: Multilingual English-German Image Descriptions (2016). Disponible en <https://arxiv.org/abs/1605.00459>.
- [17] TransformerXL from Scratch [notebook](#).