CROSS-XQUAD: APLICACIÓN DE FINE-TUNING EN MODELO EXTRACTIVO PARA UN SISTEMA DE PREGUNTAS Y RESPUESTAS MULTILENGUAJE

Leonardo Ángel, Juan Álvarez, Edinson Cáceres y Sara Sánchez
Departamento de Ingeniería de Sistemas y Computación
Universidad de los Andes

Resumen

Con el presente trabajo se pretende mejorar la capacidad de responder preguntas creando un modelo de machine learning desplegado en la nube y accedido mediante una interfaz web. Se quiere tomar un modelo pre-entrenado y mejorar la capacidad del mismo para que pueda entender un texto introducido y responda preguntas formuladas en español, independientemente del idioma del texto inicial, dando una respuesta en el idioma del contexto, para lo cuál el modelo resulta capaz.

Del mismo modo, se espera optimizar el entrenamiento y ejecución del modelo, optando por una base de datos de entrenamiento reducida, aunque con tan poca cantidad, el modelo no logra superar el desempeño del modelo pre entrenado.

Se despliega la solución en una interfaz web para que sea de fácil uso del usuario, además se realizan sugerencias para la mejora del modelo e investigación de otros modelos para seguir comprobando la hipótesis inicial, se requiere un mayor tiempo de experimentación y una capacidad de cómputo mejor.

1. Introducción

Dentro de herramientas del tipo chatbot o que emulan la interacción entre dos personas, se encuentran los modelos de preguntas y respuestas, que mediante un entrenamiento, sobre ciertos tópicos, el modelo construído puede responder las preguntas realizadas por un humano.

En el presente proyecto se pretende revisar la creación de modelos Q&A (pregunta y respuesta por sus siglas en inglés), y así mismo crear y entrenar uno propio, a partir de un modelo pre-entrenado, que sea capaz de responder preguntas en varios idiomas, además se aplicará Fine-Tuning y se analizará el comportamiento del modelo respondiendo las mismas preguntas en varios idiomas.

Además se analizarán los resultados obtenidos por el mismo modelo y se formulará una serie de conclusiones y recomendaciones con base a estos resultados. Dichas conclusiones y recomendaciones pretenden dar aproximaciones a lo que debería hacerse con la herramienta para seguir mejorando su desempeño y posibles escenarios de uso de la misma.

2. Estado del arte

Se realizó una búsqueda intensiva sobre otros modelos creados que pudiesen dar un acercamiento profundo hacia la creación, modelamiento y bases de datos existentes en los modelos de Q&A en inglés. De esta búsqueda se resaltan los conceptos claves para el desarrollo del presente trabajo, a continuación.

La arquitectura **Transformer** es la base del desarrollo de estos modelos, introducidos en 2017 revolucionaron el procesamiento de lenguaje natural, utiliza una estructura codificador - decodificador pero a diferencia de las RNN (Recurrent Neural Network) no procesan los datos secuenciados en orden, su procesamiento se basa en multiplicaciones de matrices lo cual trae los beneficios de la paralelización y tiempos de entrenamiento reducidos. Esta arquitectura utiliza la atención para enfocarse en los estados ocultos y sus cálculos no son secuenciales.

La atención se considera como una consulta, un conjunto claves-valores y una salida. La entrada consiste de claves de dimensión d_k y valores de dimensión d_v, se calcula el producto punto de la consulta con las claves se divide cada uno por $\sqrt{d_k}$ y se aplica la función softmax para obtener los pesos. La arquitectura se presenta en la parte izquierda de la figura 1. También se

puede aplicar atención multi cabeza en la cual en lugar de trabajar con una sola función de atención se trabaja con varias, lo cual permite prestar atención a diferentes subespacios de representación en diferentes posiciones, su arquitectura se muestra en la parte derecha de la figura 1.

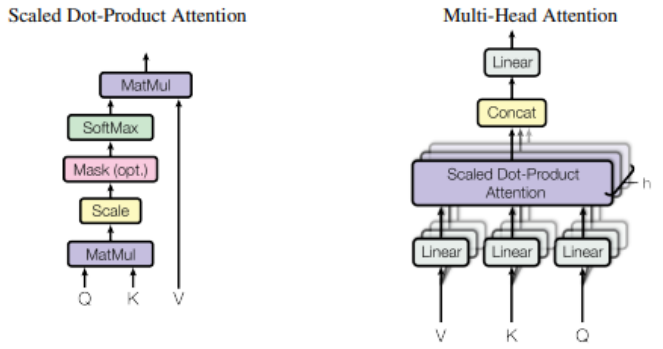


Figura 1: Atención en modelos pre-entrenados.
Fuente: Attention is all that you need, Vaswani

Devlin at. al. (2019) exploran **BERT** (bidirectional encoder representations for transformers), transformadores bidireccionales profundos, para la comprensión del lenguaje. Ya que BERT está diseñado para entrenar representaciones pre-entrenadas bidireccionales profundas a partir de texto sin etiquetar, el resultado se puede ajustar con solo una capa de salida adicional para crear modelos de última generación para una amplia gama de tareas, como la respuesta a preguntas y la inferencia de lenguaje, sin modificaciones sustanciales en la arquitectura específica de la tarea.

Mejoraron significativamente los resultados en tareas PNL, ayudando a consolidar el aprendizaje por transferencia, se basa en representaciones de lenguaje contextualizado pre entrenadas como los modelos ELMo y Fine-Tuning con parámetros mínimos específicos como en GPT.

La arquitectura BERT original utiliza bloques de codificadores y cada bloque utiliza cabeza de autoatención. Utiliza un algoritmo de tokenización de subpalabras wordpiece el cual aprende una serie de reglas de combinación iterando palabra y subpalabras tomando en cuenta las más frecuentes.

Bert tiene un enfoque bidireccional debido al modelo de lenguaje enmascarado (MLM), tarea en la cual el 15% de las palabras en cada secuencia de entrada se reemplaza con una máscara, esto ocurre durante el pre-entrenamiento en el cual además existe otro objetivo llamado predicción de la siguiente oración (NSP) que consiste en averiguar si la siguiente oración sigue de manera lógica la primera.

Para el Fine-Tuning se requiere ingresar una entrada y una salida adecuada, permitiendo actualizar los parámetros de la red, para el caso de preguntas y respuestas se relacionan pregunta-pasaje. Las representaciones de los tokens se alimentan a una capa de salida para tareas a nivel de token.

Reddy at. al (2019), por su parte, abordan el modelo y base de datos ***CoQA:*** *A Conversational Question Answering* Challenge. Parten de la premisa que para que una máquina aprenda, debe ser capaz de realizar y responder preguntas conversacionales. CoQA contiene 127.000 preguntas con respuestas, etiquetadas en 7 áreas de conversación diferentes. Se observa que el modelo tiene un buen razonamiento, obtiene f1 score de 65.4% en cuanto que el humano es de 88.8%, el modelo falla en el entendimiento de correferencias y razonamiento pragmático.

En 2020 Wang et. al., indican que aplicar **Fine-Tuning** a un modelo pre entrenado, permite obtener mejores resultados que un modelo NLP (Procesamiento de Lenguaje Natural por sus siglas en inglés). El Fine-Tuning que se propone consta de una herramienta que se centra en solamente aprender de instancias típicas de varios dominios para adquirir conocimiento altamente transferible para el modelo, esto es, no entrenar con un conjunto enorme de datos, sino aprender aquellas palabras o secuencias que se asocian a una etiqueta particular, de este modo, lograr una contextualización eficaz de una palabra en específico.

Por último **XQUAD,** que ayudará a la configuración del modelo propuesto, se trata de un conjunto de datos de respuesta a preguntas en varios idiomas, para poder evaluar el rendimiento del modelo en distintos lenguajes. Este conjunto consta de 240 párrafos, 1.190 pares de preguntas y respuestas del conjunto de desarrollo SQuAD (The Stanford Question Answering Dataset) y su traducción de alto nivel a español, inglés, rumano, alemán, griego, ruso, turco, árabe, vietnamita, tailandés, chino e hindi. Es el mismo conjunto de datos, en 12 idiomas diferentes con 9 alfabetos distintos.

3. Metodología

El modelo que se estudió en el correspondiente proyecto es BERT con un Fine-Tuning sobre XQuAD (por simplicidad nos referimos al modelo como XBERT). XQuAD es un conjunto de datos de tipo Q&A, con tuplas (*contexto, preguntas, respuestas*) en 12 idiomas diferentes. Cada una de estas tuplas tiene una correspondiente traducción en los 11 idiomas restantes realizada por diferentes expertos en el área. Es importante resaltar que para cada tupla de XQuAD el contexto, las preguntas y respuestas se encuentran en un mismo idioma.

En el experimento expuesto se evalúa la capacidad del XBERT para realizar una tarea para la que no fue precisamente entrenado: responder preguntas en español dado un contexto en cualquiera de los 12 idiomas presentes en el dataset. Cada una de las preguntas en español obtendrá respuesta en el idioma del contexto pues XBERT es un modelo extractivo, es decir, busca la respuesta de forma explícita en el párrafo contextual correspondiente. Por ejemplo, si:

Contexto: *I'm Sebas and I studied computing*
Pregunta: *What did Sebas study?*
Respuesta esperada: *computing*

4. Descripción y evaluación del experimento

Inicialmente se evaluó la capacidad del XBERT para la tarea de responder en diferentes idiomas a preguntas en español sin ningún entrenamiento previo. Posteriormente se realizó una modificación al dataset XQuAD, para obtener un dataset en el que todas las preguntas están escritas en español pero los contextos y respuestas se encuentran en cada uno de los diferentes idiomas. El entrenamiento se realizó usando la GPU de Google Colab durante una época.

Las métricas usadas para la evaluación del modelo fueron **Exact match** y **F1 score.** Exact match retorna 1 si la predicción corresponde exactamente con la respuesta esperada y 0 en otro caso. F1 score usa la relación clásica entre precisión y sensibilidad siendo la precisión el cociente entre el número de palabras compartidas y el número total de palabras de la predicción, mientras que la sensibilidad es el cociente del número de palabras compartidas sobre el número de palabras de la respuestas esperada.

5. Resultados obtenidos

Después de realizar fine tuning sobre XBERT usando 1, 2, 5, 10, 20 y 25 muestras aleatorias, se realizó la evaluación de cada uno de los modelos:

	Número de datos usados para el fine-tuning						
	0	**1**	**2**	**5**	**10**	**20**	**25**
Spanish	96.08	84.31	66.67	78.43	58.82	74.51	64.71
English	94.12	80.39	35.29	54.90	43.14	70.59	68.63
German	90.20	52.94	39.22	39.22	37.25	54.90	43.14
Arabic	76.47	43.14	27.45	33.33	19.61	41.18	31.37
Russian	74.51	62.75	27.45	35.29	23.53	47.06	47.06
Vietnamese	70.59	47.06	41.18	43.14	33.33	52.94	45.10
Chinese	68.63	39.22	23.53	29.41	31.37	45.10	43.14
Romanian	64.71	41.18	25.49	33.33	23.53	35.29	31.37
Hindi	58.82	43.14	29.41	25.49	35.29	41.18	37.25
Greek	56.86	41.18	27.45	21.57	21.57	39.22	31.37
Turkish	54.90	35.29	25.49	23.53	21.57	35.29	39.22

Tabla 1: Exact match sobre Cross-XQuAD
Fuente: Elaboración propia

	Número de datos usados para el fine-tuning						
	0	**1**	**2**	**5**	**10**	**20**	**25**
Spanish	99.00	87.06	75.36	80.69	65.42	79.32	68.95
English	95.80	81.37	42.12	60.95	49.94	73.23	73.72
German	90.69	59.16	44.11	44.14	40.51	60.83	49.19
Arabic	82.95	48.86	31.77	36.51	22.00	44.24	38.72
Russian	81.31	66.38	34.63	39.12	24.89	49.18	48.53
Romanian	77.32	49.77	35.62	43.15	28.95	46.42	40.11
Vietnamese	75.47	53.92	46.47	53.46	36.99	58.66	54.03
Chinese	71.24	44.44	27.45	34.64	35.29	49.02	49.67
Greek	68.50	48.19	35.42	26.67	27.55	44.17	38.05
Hindi	67.85	51.84	37.07	33.63	39.30	46.49	44.14
Turkish	66.04	45.64	34.10	28.81	27.56	38.65	47.65
Thai	54.30	33.71	24.23	29.46	24.36	22.55	22.92

Tabla 2: F1 score sobre cross-XQuAD
Fuente: Elaboración propia

6. Despliegue de la solución

La herramienta - modelo - creada para el presente proyecto es una aplicación frontend desplegada en Heroku y una API backend desplegada en AWS, lo que permite a cualquier usuario con el link de acceso interactuar con la herramienta.

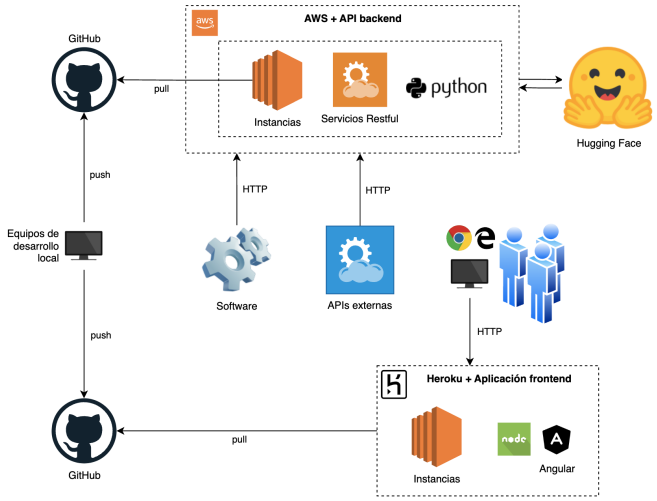
Para la configuración de este espacio web, se usaron las siguientes herramientas:

Herramientas utilizadas:

- AWS + EC2. Se utiliza una instancia tipo inf1.xlarge que está optimizada para proyectos de Machine Learning, se compone de 4 vCPUs y 8 GB RAM. Adicional tiene un volumen de 30 GB tipo gp2, esto es un disco SSD.
- GitHub como repositorio de código.
- Postman (Para pruebas locales).

Tecnologías:

- Python + Flask (Backend).
- Angular + HTML + Bootstrap (Frontend).
- NodeJS - Express (Web server frontend).
- Gunicorn (Web server backend).



Imágen 2: Creación de interfaz web.
Fuente: Elaboración propia.

El API backend se encuentra en GitHub y los cambios son descargados dentro de una instancia EC2 de AWS.

El API usará los modelos de ML que están disponibles en Hugging Face; un modelo bert-multi-cased-finetuned-xquadv1 (Que lo llamaremos "modelo base") y otro modelo bert-finetuned-crossxquadv1_25sbl (Que lo llamaremos "modelo propio"). Dentro del API no entrenaremos los modelos nuevamente, solamente hacemos uso de estos.

La aplicación frontend se encuentra en GitHub, el deploy de la aplicación se realiza de forma automática en Heroku una vez se hacen push de los cambios en la rama main.

URLs:

- API restful.
- Aplicación frontend.

7. Discusión

Los resultados muestran que para idiomas inglés y español se tiene un score f1 alto, sin embargo al hacer la prueba en la solución web se observa que las respuestas entre el modelo base y el modelo propio, en algunas ocasiones no son las mismas, dando como ganador al modelo base.

Las limitaciones en nuestra metodología son los recursos de cómputo con los que contamos ya que el entrenamiento fue demorado, también el poder utilizar más muestras en otros idiomas.

Para un trabajo futuro se recomienda migrar a un ambiente de producción más robusto para realizar mejores entrenamientos, contar con más muestras de textos y preguntas y también aumentar los ámbitos de conocimiento en las muestras escogidas.

8. Conclusiones y recomendaciones

Una vez obtenidos los resultados y generada la discusión, se puede afirmar que:

- El modelo es capaz de inferir un párrafo, una pregunta y responder, de acuerdo a la información del párrafo, independiente del idioma en el que se ingrese el párrafo o pregunta, así sean diferentes.
- La herramienta, aunque tiene un desempeño variable en las respuestas otorgadas a las preguntas realizadas, en general funciona muy

bien cuándo palabras claves dentro de la pregunta se encuentran en el texto.

- Ya que el modelo logra entender 12 idiomas, entre los cuales se incluyen diferentes alfabetos y estos cubren la mayoría del lenguaje a nivel mundial, no se considera necesario expandir la base de idiomas, añadir nuevas muestras, pero de poca información (por que serían nuevos idiomas sin la extensión y capacidad de los ya analizados) podría ser contraproducente y generar un efecto contrario en la precisión del modelo.
- El modelo fue entrenado en tiempos óptimos, sin embargo requiere una máquina medianamente robusta, pues los ambientes gratuitos ofrecidos por herramientas como Google Colab, no son suficientes para el tratamiento de la data.
- Desplegar una solución de machine learning en la nube es fácil usando herramientas como Hugging Face o Gradio, sin embargo, si se desea tener mayor autonomía y personalización de la solución hay que usar otras herramientas como Heroku o AWS.
- Los despliegues personalizables no son fáciles de realizar, hay que contar con tiempo y conocimiento de APIs, backend, redes y lenguajes de programación. Es por eso que Hugging Face o Gradio toman relevancia dado su facilidad de uso.

Para seguir mejorando el modelo, se recomienda:

- Para futuras interacciones con la herramienta, se sugiere ampliar el campo de conocimiento del modelo, aunque es capaz de responder preguntas sobre el texto introducido, sería bueno y deseable incluir nuevos párrafos para entrenar la inferencia del modelo.
- Se recomienda además, migrar a un ambiente de producción más robusto, con el fin de alimentar la base de datos de preguntas y respuestas de ejemplo, cada vez más.
- Aunque el modelo es capaz de inferir textos y entregar respuestas, se debe re evaluar y rediseñar el modelo con muchas más muestras de preguntas y respuestas en los distintos idiomas.

9. Referencias

Explorer Feda (2021). *NLP-progress:Question answering*. Tomado de: http://nlpprogress.com/english/question_answering

Devlin J., Chang M., Lee K., Toutanova K. (2019). *BERT: Pre-training of deep bidirectional transformers for language understanding.* Disponible en NAACL HLT 2019 - 2019 “Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies” - Proceedings of the Conference, 2019. ISBN 9781950737130. Tomado de: https://arxiv.org/pdf/1810.04805.pdf

Reddy S., Chen D., Manning C. (2019). *CoQA: A Conversational Question Answering Challenge.* Tomado de: https://arxiv.org/pdf/1808.07042.pdf

Wang C., Qiu M., Huang J., He X. 2020. *Meta Fine-Tuning Neural Language Models for Multi-Domain Text Mining*. Tomado de: https://arxiv.org/pdf/2003.13003v2.pdf

Vaswan A., Shazeer N., Parmar N., Uszkoreit Jones J., Gomez A., Kaiser L., Polosukhin I., (2017). *Attention is all you need*. Disponible en “Advances in Neural Information Processing Systems, vol. 2017-December”. Neural information processing systems foundation, 6 2017. ISSN 10495258 pp. 5999–6009. Tomado de: https://arxiv.org/abs/1706.03762v5

Ruder S. (2021). *XQuAD.* Tomado de: https://github.com/deepmind/xquad

Hugging Face (2018). Dataset: Scientific Papers. Recuperado el 11 de Octubre de 2022 de: https://huggingface.co/datasets/scientific_papers

The Stanford NLP Group. (2020). CoQA: A Conversational Question Answering Challenge. Recuperado el 11 de Octubre de 2022 de: https://stanfordnlp.github.io/coqa/

The Stanford NLP Group. (2022). SQuAD 2.0: The Stanford Question Answering Dataset. Recuperado el 11 de Octubre de 2022 de: https://rajpurkar.github.io/SQuAD-explorer/