Aplicaciones de las Redes Recurrentes y de la Arquitectura Transformer al Procesamiento de Lenguaje Natural

Valencia Jaramillo, Juan José. Leon Torres, Harold

Profesor: Jesús Alfonso López

Facultad de ingenieria, Universidad Autonoma de Occidente

Santiago de Cali, Colombia 10 de febrero de 2025

Abstract: Este articulo presenta un trabajo sobre Procesamiento de Lenguaje Natural (PLN) con aplicaciones en Redes Recurrentes y Arquitectura Transformer en el Dataset "HeadQA"**[1]**, un dataset con preguntas de salud para razonamiento complejo. Se muestra a detalle el resumen de la problemática y la metodología. Los resultados muestran un desempeño de 97,89 % y 98,68 % para los modelos de redes recurrentes. 37,98 % y 24,54 % para los modelos de transformers. Se sugieren oportunidades de mejora como profundización en la complejidad de los modelos y en la caracterización de los datos para capturar más contextos y detalles durante la clasificación. Adicionalmente se propone un trabajo futuro para propuestas asistidas por IA con retroalimentación o 'tutorias' que expliquen la razón de ser de una respuesta, el contexto de una pregunta, etc.

Palabras clave: Inteligencia artificial, Redes recurrentes, Transformers, PLN, Salud.

1. Introducción

Los últimos avances en la respuesta a preguntas han sido impulsados por modelos neuronales debido a su capacidad para tratar textos sin procesar [2], [3]. Sin embargo, algunos investigadores han observado una tendencia hacia el desarrollo de conjuntos de datos y métodos que se adapten a los puntos fuertes y a la naturaleza intensiva en datos de estos modelos [4], [5].

Conjuntos de datos populares como bAbI [6] y SQuAD [7],[8] han permitido a los sistemas alcanzar un rendimiento cercano al humano, que a menudo sólo requiere conocimientos superficiales para responder a las preguntas [9], [10]. Para hacer frente a esta limitación, se han introducido conjuntos de datos con múltiples opciones que requieren razonamiento. Por ejemplo, Clark et al. desarrollaron conjuntos de datos de ciencias de primaria debido a la dificultad de obtener preguntas especializadas [5], [11].

Es necesario disponer de un conocimiento especifico en los dominios de las preguntas, así como contar con la capacidad de razonar y resolver problemas. Es por eso que en este trabajo se va a usar HEAD-QA [1] para entrenar 2 modelos basados en redes recurrentes y 2 modelos basados en arquitectura transformer para realizar una tarea de clasificación multiclase supervisada, específicamente en el contexto de selección de respuestas correctas para preguntas de opción múltiple. Este tipo de tarea cae dentro del campo de PLN.

2. Metodología

2.1. Dataset

HEAD-QA [1] es un conjunto de datos multirrespuesta sobre asistencia sanitaria. Las preguntas proceden de exámenes profesionales para acceder a un puesto especializado en el sistema sanitario español que requiere de conocimientos y razonamientos profundos, y suponen un reto incluso para seres humanos que han estudiado en estos campos.

Las preguntas están diseñadas por el Ministerio de Sanidad, Consumo y Bienestar Social, que también proporciona acceso directo a los exámenes de los últimos 5 años.HEAD-QA trata de hacer estas preguntas accesibles a la comunidad del PLN. Esperamos que sea un recurso útil para conseguir mejores sistemas de control de calidad.

El conjunto de datos contiene preguntas sobre los siguientes temas: *medicina, enfermería, psicología, química, farmacología y biología.* A continuación se presentan algunos ejemplos de preguntas disponibles en HEADQA, las respuestas co-

rrectas están resaltadas en amarillo.

Pregunta (medicina): Una niña de 13 años es operada debido a la enfermedad de Hirschsprung a los 3 meses de edad. ¿Cuál delos siguientes tumores es más probable que presente?

1. Neuroblastoma abdominal
2. Tumor de Wilms
3. Nefroma mesoblástico
4. Carcinoma medular tiroideo familiar.

Pregunta (farmacología) El tratamiento antibiótico de elección para la meningitis causada por Haemophilus influenzae serogrupo b es:

1. Gentamicina
2. Eritromicina
3. Ciprofloxacino
4. Cefotaxima

Pregunta (psicología) Según las investigaciones derivadas del modelo de Eysenck, hay pruebas de que los extravertidos, en comparación con los introvertidos:

1. Rinden mejor en tareas de vigilancia.
2. Tienen mayor secreción salival antes de la prueba del limón.
3. Tienen mayor necesidad de estimulación.
4. Tienen menor tolerancia al dolor.

Teniendo en cuenta esto, cada dato especifico (pregunta) del conjunto está guardado como un JSON con la siguiente estructura:


```
{
qid": "1",
'category': 'biology',
'qtext': 'Los potenciales postsinápticos excitatorios:',
'answers': [
    {
        'ayuda': 1,
        'atext': 'Son del tipo todo o nada.'
    },
    {
        'aid': 2,
        'atext': 'Son hiperpolarizantes.'
    },
    {
        'aid': 3,
        'atext': 'Pueden sumarse.'
    },
    {
        'aid': 4,
        'atext': 'Se extienden a grandes distancias.'
    },
    {
        'aid': 5,
        'atext': 'Tienen un periodo refractario.'
    }],
'ra': '3',
'image': <PIL.PngImagePlugin.PngImageFile image mode=RGB size=675x538 at 0x1B42B6A1668>,
'name': 'Notebook_2013_1_8',
'year': '2013'
}
```


Figura 1: Ejemplo de pregunta en formato JSON

2.2. Particionamiento

Se dividió el conjunto de datos en entrenamiento, validación y pruebas. Los tamaños se muestran a continuación:

Cuadro 1: Particionamiento del dataset

	Train	Val	Test
HeadQA	2657	1366	2742

2.3. Preprocesamiento

2.3.1. Para Redes Recurrentes

Dado que Las RNNs requieren convertir el texto en secuencias numéricas comprensibles, se realizó un preprocesamiento de los datos usando un tokenizador **(tokenización)** con el objetivo de crear un vocabulario para ser usado como entrada por los modelos con un numero de palabras máximo de 10000. Esto le dice al tokenizador que sólo considere las 10.000 palabras más frecuentes en el conjunto de datos.

Para cada ejemplo, se proces la pregunta (example['qtext']), las opciones de respuesta (example['answers']) y la pregunta correcta (example['ra']). Además se aplica "paddingüna técnica de normalización que hace que todos los datos tengan el mismo tamaño, en este caso después de la tokenización los datos se pasan a vectores que representan las frases textuales y cuando estas cadenas no llegan a completar el tamaño fijo, el resto de posiciones se rellenan con ceros.

El proceso de tokenización, vectorización y normalización se realiza para todos los conjuntos de datos disponibles (train, test, validation).

2.3.2. Para Transformers

Para los transformers se carga un tokenizador predeterminado, en este caso de BERT, con esto se descarga un diccionario lingüístico para un modelo especifico, el ≪biobert-base-cased-v1.1≫. Este modelo específico es una versión de BERT que distingue entre mayúsculas y minúsculas (es decir, trata ≪Hola≫ y ≪hello≫ como palabras distintas). Esto es útil pues para la predicción que buscamos, puede ser necesario considerar palabras sensibles mayúsculas o minúsculas como nombres propios. Además, incluye en su vocabulario palabras y expresiones relacionadas al campo de la salud.

2.4. Modelos de redes recurrentes

2.4.1. Modelo 1: LSTM [12]

Esta arquitectura (ver figura 2) emplea una capa de Embedding para transformar las secuencias numéricas en representaciones vectoriales densas de 128 dimensiones, facilitando la captura de relaciones semánticas entre las palabras. A continuación, una capa LSTM con 64 unidades procesa secuencialmente la entrada, capturando dependencias temporales. El modelo solo retiene el estado oculto final, el cual es alimentado a una capa Dropout para mitigar el sobreajuste. Finalmente, una capa Dense con una función de activación softmax genera las probabilidades de pertenencia a cada clase, completando el proceso de clasificación.

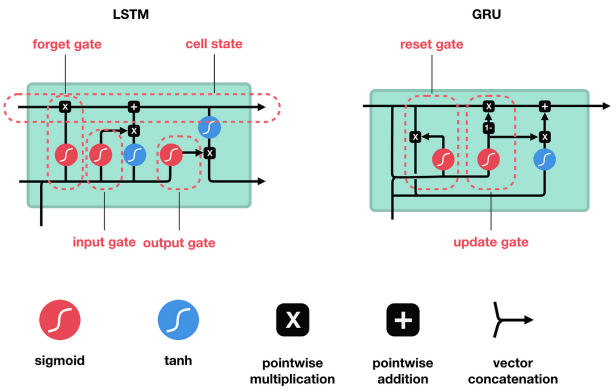

Figura 2: Unidad LSTM y GRU

2.4.2. Modelo 2: LSTM GRU

la GRU (ver figura 2) simplifica el mecanismo de aprendizaje a largo plazo al combinar las funciones de la puerta de entrada y la puerta de olvido en una sola llamada puerta de actualización, junto con una puerta de reinicio, lo que reduce la complejidad computacional y acelera el entrenamiento sin perder capacidad de modelado. Aunque ambas arquitecturas son eficaces para tareas de secuencias, la LSTM suele ser preferida en problemas que requieren capturar dependencias muy largas, mientras que la GRU ofrece un rendimiento similar con menos recursos computacionales.

2.5. Modelos de transformers

2.5.1. BioBERT [13]

BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) es una variante de BERT preentrenada con grandes corpus de texto biomédico, como PubMed y PMC. Fue desarrollada para tareas de NLP en el ámbito de la biomedicina, donde el lenguaje técnico y especializado requiere un entendimiento más profundo que el texto general (ver figura 3).

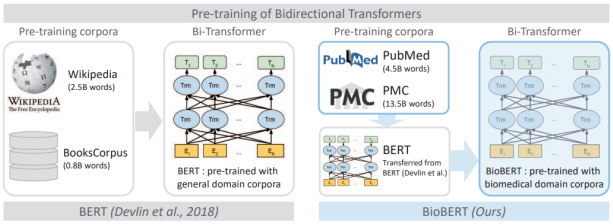

Figura 3: Arquitectura-componentes: BioBERT

2.5.2. DistilBERT

DistilBERT es una versión más ligera de BERT que utiliza una técnica llamada destilación de conocimiento (knowledge distillation). En este proceso, un modelo grande (como BERT) enseña a un modelo más pequeño a imitar su comportamiento, reduciendo así el tamaño del modelo sin perder demasiada precisión (ver figura 4).

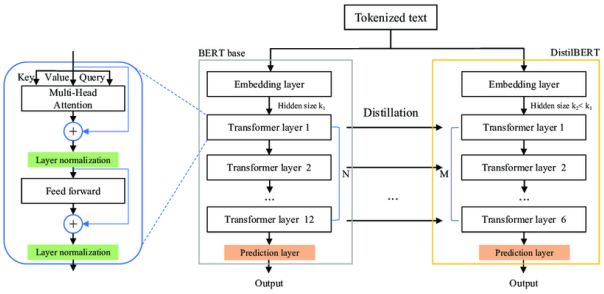

Figura 4: Arquitectura DistilBERT

3. Resultados

Todo el código fue implementado en Google Colab con una máquina T4-GPU: 1.59 Ghz, 40 nucleos, 16 Gb de memoría. Aquí se presentan las gráficas del proceso de entrenamiento de los 4 modelos sus respectivas matrices de confusión.

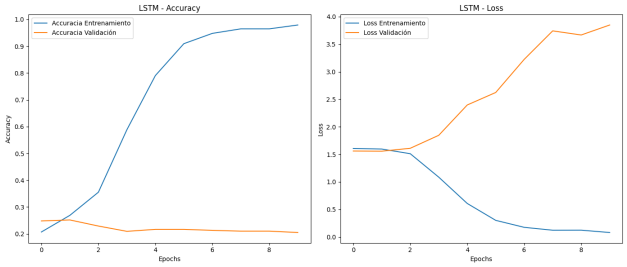

Figura 5: Accuracy-Loss en entrenamiento: LSTM

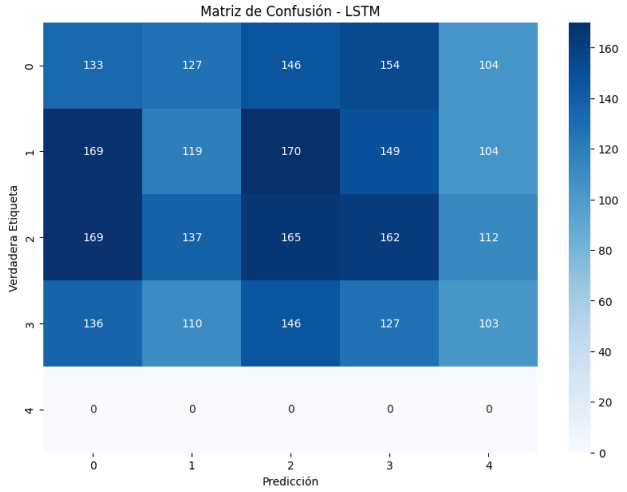

Figura 6: Matriz de confusión: LSTM

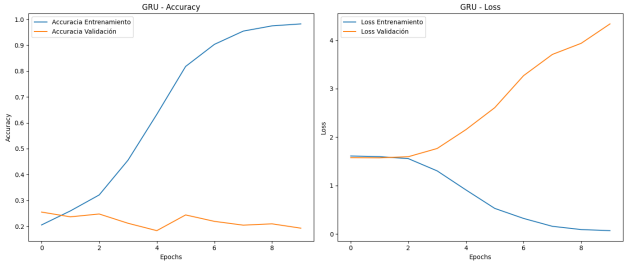

Figura 7: Accuracy-Loss en entrenamiento: GRU

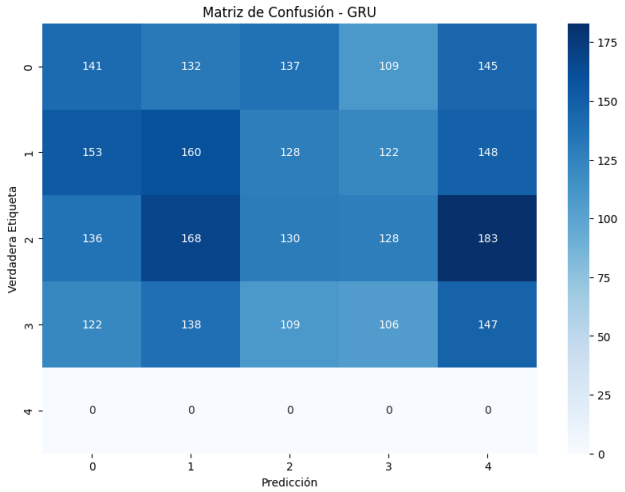

Figura 8: Matriz de confusión: GRU

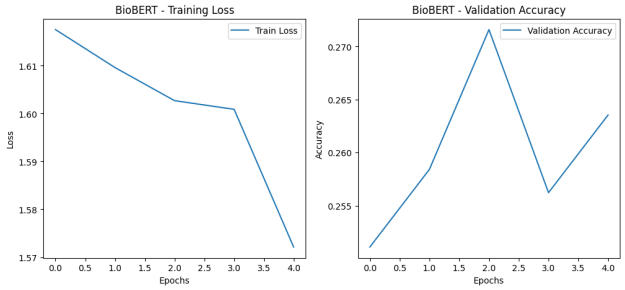

Figura 9: Accuracy-Loss: BioBERT

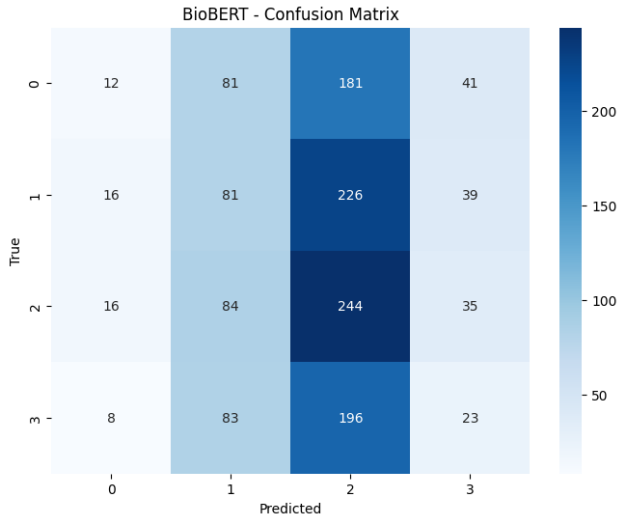

Figura 10: Matriz de confusión: BioBERT

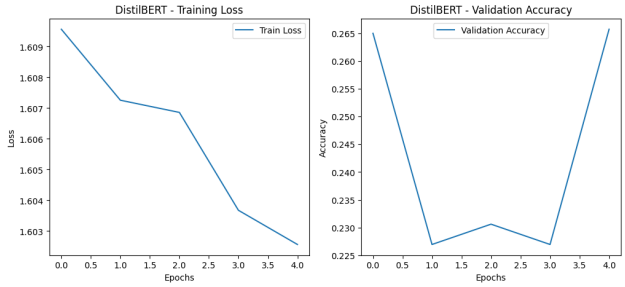

Figura 11: Accuracy-Loss: DistilBERT

3.1. Modelo 1: LSTM simple

3.1.1. Parámetros

LR=0.001,Epochs=10,Optimizer=Adam, BatchSize=16,Dropout=0.5,activation=SoftMax.

3.1.2. Resultados

Precisión en el conjunto de entrenamiento:0.9789 Precisión en el conjunto de prueba:0.1984

3.2. Modelo 2: LSTM bidireccional

3.2.1. Parámetros

LR=0.001,Epochs=10,Optimizer=Adam, BatchSize=16,Dropout=0.5,activation=SoftMax.

3.2.2. Resultados

Precisión en el conjunto de entrenamiento:0.9868 Precisión en el conjunto de prueba:0.1958

3.3. Modelo 1: BioBERT

3.3.1. Parámetros

LR=0.00002,Epochs=5,Optimizer=Adam, BatchSize=16.

3.3.2. Resultados

BioBERT Accuracy - Train: 0.3798, Val: 0.2635, Test: 0.2553

3.4. Modelo 2: DistilBERT

3.4.1. Parámetros

LR=0.00002,Epochs=5,Optimizer=Adam, BatchSize=16.

3.4.2. Resultados

DistilBERT Accuracy - Train: 0.2454, Val: 0.2657, Test: 0.2626

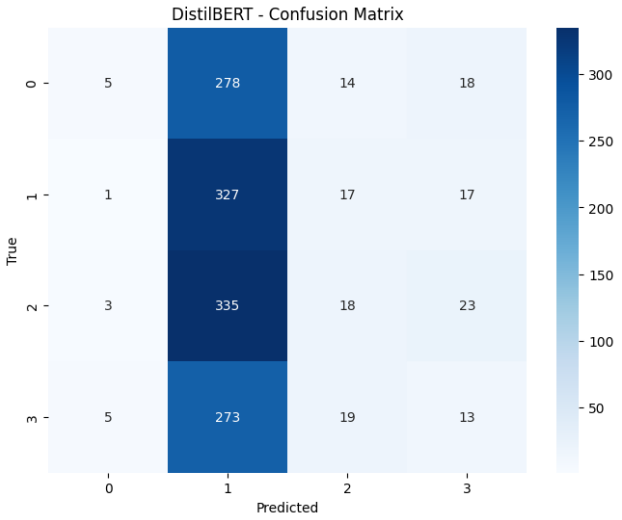

Figura 12: Matriz de confusión: DistilBERT

4. Analísis de resultados

Tras haber llevado a cabo el entrenamiento y después de haber revisado los resultados es apropiado decir que existen algunos fallos y oportunidades de mejora para el trabajo con el fin de lograr un mejor rendimiento.

4.0.1. Fallos generales

Sobreajuste: Es un problema común en todos los modelos. Esto se manifiesta en un alto rendimiento en el conjunto de entrenamiento pero un rendimiento mucho menor en el conjunto de validación y prueba. El modelo aprende patrones muy específicos de los datos de entrenamiento que no generalizan bien a datos nuevos.

Bajo rendimiento en general: Todos los modelos tienen dificultades para predecir correctamente las clases, lo que se refleja en una baja precisión general y una alta confusión entre clases en las matrices de confusión. Esto sugiere que las características que distinguen las diferentes clases no están siendo capturadas de manera efectiva por los modelos.

5. Conclusiones

El PLN es un campo inherentemente complejo debido a la naturaleza ambigua y matizada del lenguaje humano. En datos como los de HeadQA, donde se busca comprender preguntas y respuestas para seleccionar la opción correcta, los modelos de NLP deben ser capaces de capturar no

solo el significado individual de las palabras, sino también las relaciones semánticas y contextuales entre ellas.

HeadQA presenta desafíos que requieren un enfoque más holístico para el modelado de PLN. No basta con procesar preguntas y respuestas de forma independiente; es crucial modelar la interacción entre ellas para determinar la respuesta correcta.

Nuestros resultados indican que los modelos que hemos probado no son capaces de capturar completamente esta interacción. Las matrices de confusión muestran una alta confusión entre clases, lo que sugiere que los modelos no están aprendiendo a distinguir entre las diferentes opciones de respuesta de manera efectiva.

Para abordar estos desafíos, es necesario considerar una combinación de estrategias, que incluyen:

Mejorar la arquitectura del modelo: Explorar arquitecturas más complejas que modelen explícitamente las relaciones entre preguntas y respuestas, como modelos de ”matching.º .ªttention”.

Aumentar la cantidad y calidad de los datos: Disponer de más datos de entrenamiento y mejorar la calidad de las anotaciones puede ayudar a los modelos a aprender patrones más robustos y generalizables.

Ajustar los hiperparámetros y la estrategia de entrenamiento: Optimizar los hiperparámetros y utilizar técnicas de regularización y detención temprana puede mejorar la convergencia y el rendimiento de los modelos.

Considerar el conocimiento externo: Incorporar conocimiento externo, como bases de datos de conocimiento o información semántica, puede ayudar a los modelos a comprender mejor el contexto de las preguntas y respuestas.

6. Trabajo futuro

Dada la complejidad del conjunto de datos, sería adecuado proponer otras aplicaciones para maximizar la utilidad de la información.

6.1. Evaluación educativa y retroalimentación:

Para desarrollar sistemas de calificación automatizados que evalúen la calidad de las respuestas de los estudiantes a preguntas de nivel universitario. Las preguntas podrían utilizarse para crear pruebas que no sólo evalúen la corrección de las respuestas, sino también la profundidad de la comprensión y el proceso de razonamiento. Los sistemas podrían proporcionar información detallada a los estudiantes, destacando las áreas de mejora en su forma de resolver problemas.

6.2. Tutoría asistida por IA:

En el ámbito educativo, los sistemas que utilicen este conjunto de datos podrían servir de tutores a los estudiantes que se enfrenten a materias complejas de posgrado. Esto podría guiar a los estudiantes hacia las respuestas correctas y también les podría ayudar a entender el razonamiento que hay detrás de esas respuestas. Por ejemplo, si un estudiante está atascado en un problema, basado en las preguntas se podrían ofrecer pistas o sugerir conceptos y metodologías relevantes.

Referencias

[1] David Vilares and Carlos Gómez-Rodríguez. 2019. HEAD-QA: A Healthcare Dataset for Complex Reasoning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 960–966, Florence, Italy. Association for Computational Linguistics.

[2] Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Ha- jishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In Conference on Computer Vision and Pattern Recognition (CVPR), pages 4999–5007.

[3] Souvik Kundu and Hwee Tou Ng. 2018. A nil-awareanswer extraction framework for question answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4243–4252, Brussels, Belgium. Association for Computational Linguistics.

[4] Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In Procee-

dings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

[5] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. arXiv preprint arXiv:1803.05457.

[6] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards AI-complete question answering: A set of prerequisite toy tasks. arXiv preprint arXiv:1502.05698.

[7] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. pages 784–789.

[8] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392.

[9] Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18).

[10] Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dcn+: Mixed objective and deep residual coattention for question answering. arXiv preprint arXiv:1711.00106.

[11] Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In Proceedings of AAAI.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Comput. 9, 8 (November 15, 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[13] Lee, Jinhyuk and Yoon, Wonjin and Kim, Sungdong and Kim, Donghyeon and Kim, Sunkyu and So, Chan Ho and Kang, Jaewoo. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. 4, 36, 1234–1240