Spoiler de clickbaits con Question Answering

Unai Salaberria Flaño
usalaberria002@ikasle.ehu.eus

Abstract

Este informe muestra los procedimientos utilizados para resolver la tarea de "Spoiler de clickbaits", y los resultados obtenidos. El objetivo es aplicar las diferentes técnicas vistas a lo largo de la asignatura de Minería de Datos Textuales. Veremos la comparativa entre diferentes modelos para tareas de clasificación y question answering.

1 Introducción

Para entender el objetivo de este trabajo, primero debemos entender qué es un "clickbait" y por qué queremos crear un "spoiler" para cada uno de ellos. Un clickbait es una técnica de redacción que consiste en crear encabezados y descripciones sensacionalistas en un enlace. En la figura 1 podemos ver un ejemplo de un clickbait en la red social Twitter, que pretende captar la atención del usuario y llevarle a hacer click en el artículo del enlace. El clickbait se considera molesto ya que, normalmente, la respuesta a la duda creada suele ser trivial.

Es por eso que últimamente se han desarrollado varias técnicas para resolver esa curiosidad generada antes de entrar al artículo. En la figura 2 se puede ver un ejemplo de lo que sería el spoiler del clickbait visto antes en la figura 1.

Dicho esto, el objetivo de este trabajo es explorar varias técnicas de generación de spoiler. Además, antes de realizar esta tarea, realizaremos la clasificación de cada uno de los spoilers necesarios.

Figure 1: Ejemplo de un clickbait

Figure 2: Ejemplo de un spoiler al clickbait anterior

2 Metodología

2.1 Conjunto de datos

El conjunto de datos que vamos a utilizar para esta tarea de "clickbait spoiling" es el WebisClickbait Spoiling Corpus 2022, que pertenece a una competición creada por los autores del siguiente artículo (Hagen et al., 2022). Tiene 5000 instancias repartidas de la siguiente forma: 3200 de entrenamiento, 800 de validación y 1000 de test.

Para cada instancia del conjunto de datos tenemos la siguiente información (o variables): uuid, postId, postText, postPlatform, targetParagraphs, targetTitle, targetDescription, targetKeywords, targetMedia, targetUrl, provenance, spoiler, spoilerPositions y tags.

En la tabla 1 podemos ver cómo se distribuyen los distintos conjuntos de datos (entrenamiento, validación y test) en base al tipo de spoiler.

	Frase	Pasaje	Multi
Entrenamiento	1367	1274	559
Validación	335	322	143
Test	423	403	174

Table 1: Distribución de cada conjunto

2.2 Tarea 1: Clasificación de spoiler

La primera tarea que debemos realizar es clasificar el tipo de spoiler que vamos a necesitar para un clickbait. Hay tres tipos de spoiler posibles: de frase, de pasaje o multi. En el spoiler de frase simplemente hay que extraer una palabra o frase del artículo. En el spoiler de pasaje hacen falta varias frases del artículo para generar el spoiler. Por último, el spoiler multi necesita de varias frases no consecutivas del artículo, y por tanto, será más difícil de clasificar y generar.

2.2.1 Aproximaciones utilizadas

Para la clasificación del tipo de spoiler he creado una función en la que se recibe como parámetro el conjunto de datos, y se procesa para obtener un nuevo conjunto de datos con una sola variable predictora y el label. Esta variable predictora ('text') contiene información que puede ser relevante a la hora hacer la clasificación de texto. En principio, esta variable solamente incluía el título, la plataforma de publicación y la página web del artículo, además de una expresión regular que intenta detectar aquellas publicaciones que requieren de un spoiler de tipo multi.

Pero vistos los resultados, decidí incluir también el contexto sobre el que hay que buscar para intentar mejorarlos. Aunque no consiguiera una gran mejora esta aproximación parece más lógica, por lo que decidí continuar los experimentos de esta forma.

Realicé también una prueba en la que incluía en la variable predictora 'text' información sobre personas, organizaciones, lugares... Para ello, me basé en la aproximación de (Kruff and Tran, 2023), en la que se utiliza la librería Spacy para Name Entity Recognition. Los primeros resultados no fueron buenos, así que descarté la idea antes de seguir con la tarea.

Para terminar con las aproximaciones, he probado a entrenar los modelos sobre un conjunto que concatena los datos de entrenamiento y de validación, esperando obtener un mayor aprendizaje. Pero la idea no mejoró los resultados anteriores, así que volví al entrenamiento sobre el conjunto de entrenamiento únicamente.

2.2.2 Modelos utilizados

Para la tarea de clasificación he buscado distintos modelos de "text classification". La implementación de estos la he realizado con la librería simpletransformers, que se basa en la librería Transformers de Hugginface y facilita el entrenamiento y la evaluación de modelos basados en Transformers.

Los diferentes modelos los he entrenado con los siguientes parámetros: el número de épocas es de 5, el tamaño máximo de la secuencia es de 200, la tasa de aprendizaje es de 1e-5 y el tamaño del batch lo he dejado en su valor por defecto de 8 (ya que el modelo se entrenaba rápido, por lo general). Para determinar estos hiperparámetros he probado diferentes valores siempre sobre un mismo modelo, y luego lo he evaluado sobre el conjunto de validación. Aunque parezca que se puedan mejorar los resultados con más épocas, y por tanto más entrenamiento, a partir de la quinta época el modelo se estanca e incluso empeora los resultados.

Los modelos que he probado en esta tarea son los siguientes: RoBERTa (tanto su versión base como large), DistilBERT-base, BERT-large, Electra-base, BERTweet y XLM-RoBERTa-large. El objetivo era comparar el desempeño de distintos modelos Transformers, y ver si los resultados de alguno de ellos son lo suficientemente buenos como para utilizarlo en la próxima tarea.

2.2.3 Métodos de evaluación

A la hora de evaluar los modelos he utilizado las siguientes métricas: accuracy, f1, recall y precission. Además, he querido comprobar el accuracy para cada una de las tres clases (frase, pasaje y multi), ya que de esta forma podemos comprobar cómo de bien clasifica el modelo cada una de ellas.

2.3 Tarea 2: Generación del spoiler

La segunda tarea consiste en generar el spoiler del artículo, y la aproximación utilizada está basada en técnicas de "question answering" extractivo.

2.3.1 Aproximaciones utilizadas

Existen dos posibles aproximaciones que podemos tomar para llevar a cabo la tarea de generación del spoiler:

1) Utilizar el modelo con mejores resultados para la tarea anterior, y en base a su clasificación del

tipo de spoiler utilizar un modelo de Question Answering (para los spoilers de frase y multi) o un modelo de Passage Retrieval (para los spoilers de pasaje y multi).

2) Si no conseguimos unos resultados lo suficientemente buenos en la tarea de clasificación, generamos el spoiler directamente con un modelo de Question Answering, sin tener en cuenta su etiqueta.

Question Answering

La tarea de generación del spoiler se puede ver como un sistema de preguntas y respuestas, donde la pregunta es la publicación con el clickbait (ver ejemplo de la figura 1), y se toma el artículo de referencia como texto donde encontrar la respuesta.

Entre los modelos existentes para la tarea de Question Answering encontramos la familia de BERT, y también algunos modelos que han sido entrenados sobre el conjunto de datos SQuAD. Este conjunto contiene preguntas realizadas por humanos sobre artículos de la Wikipedia, y la respuesta es un segmento del artículo.

Dentro del Question Answering existen dos enfoques diferentes: el extractivo y el abstractivo. El QA extractivo se basa en, dada la pregunta y el contexto, el modelo extrae la respuesta de este último. En cambio, en el QA abstractivo el modelo genera una respuesta a partir del entendimiento de la pregunta y el contexto. Este segundo enfoque es muy útil cuando hay que generar respuestas que no están de forma explícita en el contexto.

En la descripción de la tarea se da a entender que conviene utilizar el enfoque extractivo del QA, por la forma en la que se ha creado el conjunto de datos. Esta será la aproximación que voy a utilizar principalmente, pero no descarto realizar experimentos con el enfoque abstractivo, ya que puede ser ventajoso para ciertos datos, como los que requieren del tipo de spoiler multi.

Passage Retrieval

En caso de utilizar la primera aproximación, es necesario utilizar técnicas de Passage Retrieval (o recuperación de pasajes). Esta tarea consiste en encontrar los pasajes más relevantes dentro de un texto para una consulta concreta.

Algunos modelos conocidos para Passage Retrieval son MonoBERT, MonoT5 y BM25.

2.3.2 Modelos utilizados

Una vez presentadas las tareas, cabe destacar que, como veremos más adelante, los resultados de la clasificación del spoiler no han sido muy buenos. Por lo menos no han sido tan buenos como para utilizar la clasificación con una certeza grande. Así que he tomado la decisión de enfocarme solamente en la generación del spoiler mediante Question Answering. Además, en (Hagen et al., 2022) se demuestra que, por mucho que la clasificación de los spoilers fuese perfecta, el enfoque Question Answering es mejor solución que Passage Retrieval en aquellos de tipo pasaje.

Antes de entrenar los diferentes modelos, he tenido que preprocesar el conjunto de datos. Primero, he quitado todas las instancias que requieren de QA abstractivo del conjunto de entrenamiento, ya que los modelos que he entrenado son de QA extractivo. A la hora de realizar la predicción, si el modelo se topa con una instancia de este tipo no devolverá respuesta. Además, he tenido que darle un formato concreto a cada instancia del conjunto de datos. Me he basado en la estructura del conjunto SQuAD, donde cada instancia es un diccionario con los siguientes elementos: 'title', 'context', 'qas', 'id' y 'answers'.

Los parámetros para el entrenamiento los he establecido entrenando el modelo RoBERTa-base sobre un conjunto reducido de entrenamiento y evaluándolo sobre el conjunto de validación. Una vez escogidos, he realizado el entrenamiento y el test de los modelos.

También para esta tarea he querido comparar el desempeño de distintos modelos. Por eso, he entrenado y evaluado los siguientes: RoBERTa-base, RoBERTa-base-squad, BERT-base-uncased, ALBERT-base-v2, xlm-RoBERTa-base, Electra-base y DistilBERT-base.

2.3.3 Métodos de evaluación

Para evaluar los modelos de Question Answering he utilizado las siguientes métricas:

- BLEU: Realiza la comparación entre los n-gramas de las predicciones y las soluciones reales.
- Exact matching: Proporción de las predicciones que coinciden exactamente con alguna solución.
- BertScore: Similaridad entre cada token de la predicción y cada token de la solución.

Modelo	**Acc. general**	Acc. frase	Acc. pasaje	Acc. multi	**Precision**	**Recall**	**F1**
RoBERTa-base	0.74	0.76	0.75	0.63	0.74	0.74	0.73
RoBERTa-large	**0.75**	**0.78**	**0.80**	0.56	**0.76**	**0.75**	**0.75**
DistilBERT-base	0.68	0.68	0.74	0.55	0.69	0.68	0.68
BERT-large	0.71	0.69	**0.80**	0.56	0.72	0.71	0.71
Electra-base	0.71	0.77	0.72	0.52	0.71	0.71	0.70
xlm-RoBERTa-large	0.73	0.75	0.74	**0.66**	0.73	0.73	0.72

Table 2: Resultados de los distintos modelos sobre la tarea de clasificación del spoiler

Cada métrica de las mencionadas será más útil en según qué situaciones. Por ejemplo, la métrica de Exact Matching la he implementado yo mismo para comprobar cómo de bien realiza el modelo la extracción de la respuesta en el caso de los spoilers de tipo frase.

3 Resultados

3.1 Tarea 1: Clasificación de spoiler

En la tabla 2 podemos ver los resultados obtenidos por los distintos modelos en la etapa de evaluación. Todas las evaluaciones se han realizado sobre el conjunto de test completo, sin evitar aquellas instancias de tipo multi.

El modelo RoBERTa-large es el que mejor rendimiento da, pero sin mejorar mucho los resultados del mismo modelo en su versión base. Esto es interesante, ya que si requerimos de un modelo más simple y fácil de entrenar, podemos obtar por el segundo y obtener unos resultados semejantes.

Además, es curioso ver que el modelo que mejor resultado obtiene para el tipo de spoiler multi es el xlm-RoBERTa-large, que en principio es una versión multilingüe de RoBERTa-large. Podríamos sacar como conclusión que este es el modelo que mejor realiza la tarea en general. Es decir, si el conjunto de test estuviese completamente balanceado (un tercio de instancias por cada label), seguramente obtendría el mejor resultado entre los modelos utilizados.

En definitiva, el resultado obtenido por el modelo RoBERTa-large es muy bueno, ya que supera ese máximo de 0.74 en accuracy obtenido en la competición de SemEval 2023. He realizado varios experimentos, ya mencionados antes, como el de NER o el del conjunto uniendo entrenamiento y validación, pero no han mejorado los resultados.

3.2 Tarea 2: Generación del spoiler

Como ya he mencionado anteriormente, he decidido centrarme en resolver esta tarea de generación del spoiler mediante Question Answering extractivo. Entonces, para entrenar cada uno de los modelos, he quitado aquellas instancias que requieren de QA abstractivo, ya que los modelos no serán capaces de aprender nada útil de ellas. Después, en las fases de evaluación y de test, los modelos no serán capaces de realizar la predicción cuando no encuentren la respuesta a la pregunta de forma explícita en el contexto. Para estas instancias la predicción quedará vacía. Según he podido comprobar, el número de instancias dentro del conjunto de test que requieren una respuesta abstractiva, y por tanto la predicción está vacía, es 21. Esto representa un 2% del conjunto de test completo, por lo que el error debido a esto no es apenas significativo.

Las métricas de evaluación que he utilizado para comprobar el rendimiento de los distintos modelos son BLEU, exact matching y BertScore, en el apartado 2.3.3. explico más en detalle cómo funciona cada una de estas.

En la tabla 3 he recogido los resultados de los distintos modelos entrenados sobre las métricas ya mencionadas.

El modelo que mejor rendimiento ha tenido en la métrica de BLEU es la versión fine-tuned de RoBERTa-base. El fine-tuning se ha realizado con el conjunto de datos SQuAD, que es el conjunto más relevante dentro de la tarea de Question Answering. Podemos ver que la mejora sobre el mismo modelo sin fine-tuning es bastante grande. Por las características de la tarea es posible que los modelos RoBERTa-large y su versión fine-tuned hubieran conseguido mejores resultados, pero no he podido entrenarlos por limitaciones de RAM en la plataforma Kaggle.

Que el mejor modelo fuese el fine-tuned RoBERTa-base era de esperar, pero me ha sorprendido el resultado obtenido en la métrica de BertScore por el modelo Electra-base, que ha superado el de todos los demás modelos.

Aprovechando que he guardado las predicciones

Modelo	BLEU	Exact matching	BertScore
RoBERTa-base	0.206	0.174	0.639
RoBERTa-base-squad	**0.266**	**0.178**	0.671
BERT-base-uncased	0.186	0.098	0.675
ALBERT-base-v2	0.04	0.07	0.36
xlm-RoBERTa-base	0.167	0.118	0.618
Electra-base	0.214	0.128	**0.728**
DistilBERT-base	0.1	0.08	0.5

Table 3: Resultados de los distintos modelos sobre la tarea de generación del spoiler

Features	Value
postText	'This is what happens when you leave a hotel cleaner a $500 tip'
context	'(...) The video below shows the stunned cleaner initially refusing to accept the tip, before another hotel worker reassured her by saying, "You deserve it."'
spoiler	'The video below shows the stunned cleaner initially refusing to accept the tip, before another hotel worker reassured her by saying, "You deserve it."'

Table 4: Instancia del conjunto de validación

realizadas por todos los modelos sobre el conjunto de test, he decidido comparar algunas de ellas y ver cómo las evalúan nuestras distintas métricas.

Para el ejemplo que voy a comentar, he encontrado una predicción incompleta por parte del modelo Electra. La instancia que hay que predecir está en el conjunto de validación. En la tabla 4 podemos ver la información de esta instancia.

La respuesta de Electra es la siguiente: 'The video below shows the stunned cleaner initially refusing to accept the tip,'. Es una buena respuesta al clickbait, pero es incompleta. En cambio, RoBERTa-squad da la respuesta perfectamente. Por eso, he utilizado este caso para comprobar como penalizan este error cada una de las métricas.

La métrica que he desarrollado yo sería demasiado pesimista, ya que daría 1 a la predicción de RoBERTa y 0 a la predicción de Electra, lo que parece una penalización excesiva. En cambio, BLEU score da valor 1 a RoBERTa y 0.51 a Electra. Esto parece más lógico, ya que la respuesta no es mala, pero falta información importante. Por último, BertScore le da 1 a RoBERTa y 0.94 a Electra.

Después de realizar este experimento podemos asumir que BertScore es una métrica más optimista que BLEU, y dará puntuaciones más altas por lo general. Volviendo a la tabla de resultados, se puede entender que este no será el único caso en el que Electra realiza predicciones buenas pero incompletas, y por tanto, consigue un BertScore superior a los demás. Por contrario, con BLEU score obtiene resultados bastante peores en comparación con RoBERTa-squad, ya que es una métrica que penaliza más las respuestas incompletas.

4 Conclusiones

Este proyecto ha servido para explorar distintas técnicas del mundo del Procesamiento de Lenguaje Natural. En general, he querido centrarlo en la comparativa de distintos modelos para cada una de las tareas, y evaluarlos con varias métricas, para poder sacar conclusiones de cada uno de ellos.

En la tarea 1, los resultados han sido muy buenos en comparación con los de la competición, y además he podido realizar distintos experimentos con algunas técnicas como el NER (Name Entity Recognition) para tratar de mejorarlos.

En la tarea 2, no he podido conseguir resultados tan buenos como en la competición, aunque los modelos pequeños que he podido entrenar han demostrado tener entendimiento de la tarea en muchos de los casos del conjunto de datos.

Como posible mejora del proyecto, creo que se podría explorar sobre las técnicas de Question Answering abstractivo, ya que pienso que se pueden adaptar mejor a muchos casos en los que la respuesta no aparece de forma explícita y hay que entender bien el contexto. Además, en el hipotético caso de que un modelo de clasificación de texto pudiese mejorar los resultados obtenidos en la tarea 1, quizás se podría aplicar para esta segunda tarea y utilizar diferentes técnicas (Passage Retrieval) teniendo en cuenta el tipo de spoiler.

References

Matthias Hagen, Maik Fröbe, Artur Jurk, and Martin Potthast. 2022. Clickbait spoiling via question answering and passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7025–7036, Dublin, Ireland. Association for Computational Linguistics.

Andreas Kruff and Anh Huy Tran. 2023. Billie-newman at SemEval-2023 task 5: Clickbait classification and question answering with pre-trained language models, named entity recognition and rule-based approaches. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 1542–1550, Toronto, Canada. Association for Computational Linguistics.