Transformers

Notebook: github

Javier Díaz Machado

Contenido

1 (3 Puntos) Implementación práctica:

Implementa un modelo Transformer sencillo desde cero utilizando un marco de programación de tu elección (por ejemplo, PyTorch o TensorFlow). Explica las decisiones de diseño tomadas durante la implementación, incluyendo:

♣ Elección de hiperparámetros
♣ Estructura de la red,
♣ Funciones de activación utilizadas.

Elección de hiperparámetros

- **SRC_VOCAB_SIZE = 1000 / TGT_VOCAB_SIZE = 1200**: Definen el número de palabras únicas para los vocabularios de origen y destino. Estos tamaños (1000/1200) son adecuados para un modelo de demostración o un dataset con un léxico acotado, manteniendo manejables las capas de embedding.
- **D_MODEL = 256**: Es la dimensión de los embeddings y de las representaciones internas. 256 es un valor equilibrado para modelos base, ofreciendo buena capacidad sin ser excesivamente costoso computacionalmente. Es divisible por NUM_HEADS (8), lo cual es necesario.
- **NUM_HEADS = 8**: Número de cabezas en la atención multi-cabeza. 8 es una configuración estándar que permite al modelo capturar diferentes tipos de relaciones contextuales de forma paralela, dividiendo la carga de D_MODEL.
- **NUM_ENCODER_LAYERS = 3 / NUM_DECODER_LAYERS = 3**: Profundidad de las pilas del encoder y decoder. 3 capas permiten aprender transformaciones de datos suficientemente complejas para un modelo "sencillo", sin la sobrecarga de modelos más profundos.
- **D_FF = 512**: Dimensión de la capa oculta en las redes Feed-Forward. Un valor de 512 (2 * D_MODEL) proporciona una expansión interna que aumenta la capacidad no lineal del modelo en cada bloque Transformer.
- **MAX_SEQ_LEN = 80**: Longitud máxima de secuencia para la codificación posicional. 80 tokens es una longitud razonable para frases, asegurando que el modelo puede manejar secuencias de este tamaño.
- **DROPOUT = 0.1**: Tasa de dropout para regularización. Un 10% es un valor común y efectivo para mitigar el sobreajuste en diversas partes del modelo.
- **PAD_IDX = 0**: Índice para el token de padding. Usar 0 es una convención para que el modelo identifique y enmascare estos tokens, permitiendo el procesamiento de batches con secuencias de longitud variable.

Estructura de la red

1. **Encoder (Codificador):**
    - Función: "Entiende" la frase de entrada.
    - Componentes Clave por Capa:
        - Embeddings + Positional Encoding: Convierte palabras en vectores y les da información de su posición.
        - Multi-Head Self-Attention: Permite que cada palabra "mire" a las otras palabras de la *misma frase* para capturar contexto.
        - Feed-Forward Network: Procesa la salida de la atención.
    - Se apilan varias de estas capas.
2. **Decoder (Decodificador):**
    - Función: Genera la frase de salida, palabra por palabra.
    - Componentes Clave por Capa:
        - Masked Multi-Head Self-Attention: Similar al encoder, pero "enmascarada" para que al generar una palabra, solo pueda atender a las palabras anteriores de la *salida*.
        - Encoder-Decoder Attention: Permite que el decoder "mire" la salida del encoder (la frase de entrada procesada) para decidir qué generar.
        - Feed-Forward Network: Procesa la salida de la atención.
    - También se apilan varias de estas capas.

2 (1 Punto) Comparación de modelos

Compara los modelos Transformer con los modelos tradicionales de procesamiento del lenguaje natural (NLP), como las redes neuronales recurrentes (RNNs) y las redes neuronales convolucionales (CNNs). Identifica las principales ventajas y desventajas de los Transformers en comparación con estos modelos más antiguos.

Ventajas Principales de los Transformers:

- Paralelización y Eficiencia en Entrenamiento: A diferencia del procesamiento secuencial de las RNNs, los Transformers pueden procesar todos los tokens de una secuencia en paralelo, acelerando el entrenamiento.
- Mejor Captura de Dependencias a Largo Plazo: La auto-atención permite modelar relaciones entre palabras distantes de forma más efectiva que las RNNs (que pueden "olvidar") o las CNNs (enfocadas en contexto local).

- Rendimiento Superior y Transferencia de Aprendizaje: Han establecido nuevos estados del arte en múltiples tareas de NLP, y modelos pre-entrenados (como BERT, GPT) son altamente efectivos para el aprendizaje por transferencia.

Desventajas Principales de los Transformers:

- Costo Computacional Cuadrático con la Longitud de Secuencia: La auto-atención estándar es costosa para secuencias muy largas $O(n^2)$.
- Gran Necesidad de Datos: Para alcanzar su máximo potencial, especialmente entrenados desde cero, requieren grandes cantidades de datos.
- Ausencia Inherente de Información Posicional: Requieren "codificaciones posicionales" explícitas, ya que la arquitectura base no captura el orden de las palabras.

3 (1 Punto) Atención y su impacto

Explica el mecanismo de atención en los modelos Transformer y su importancia para el procesamiento del lenguaje natural.

Proporciona un ejemplo práctico de cómo el mecanismo de atención ayuda a mejorar el rendimiento en una tarea específica de NLP.

El mecanismo de atención permite a los modelos Transformer ponderar dinámicamente la importancia de diferentes partes de la secuencia de entrada al procesar cada elemento. Es como si el modelo "prestara atención" selectivamente a las palabras más relevantes para entender el contexto de una palabra específica.

Funcionamiento: Para cada palabra, se usan vectores de Consulta, Claves y Valores. La similitud entre la Query de la palabra actual y las claves de todas las demás palabras determina unos pesos. Estos pesos se usan para crear una suma ponderada de los Valores, generando una representación contextualizada.

Por ejemplo, considera traducir: "The animal didn't cross the street because it was too tired."

Al traducir "it", el mecanismo de atención ayuda al modelo a identificar que "it" se refiere a "animal". Lo hace asignando una alta "puntuación de atención" a "animal" cuando procesa "it". Esto asegura que "it" se traduzca correctamente con el género y número apropiados (ej. "el animal" -> "él estaba cansado"). Sin atención, capturar esta relación sería mucho más difícil, especialmente en frases largas.

4 (1 Punto) Exploración de modelos preentrenados

Investiga un modelo Transformer preentrenado, como BERT, GPT o T5.

- Describe cómo se entrenó el modelo.
- Indica cómo ajustarlo (fine-tune) para tareas específicas.
- Discute cómo los modelos preentrenados han impactado en el campo de NLP.
- Proporciona ejemplos de aplicaciones prácticas.

1. Entrenamiento del Modelo (Pre-entrenamiento):

BERT se entrena con una cantidad masiva de texto sin etiquetar (ej. Wikipedia). Su objetivo es entender el lenguaje de forma general antes de tareas específicas. Lo hace con dos tareas principales:

- Masked Language Model (MLM): Se ocultan palabras al azar en las frases y BERT tiene que predecir cuáles eran, usando el contexto de las palabras de alrededor (tanto antes como después, por eso es "bidireccional"). Esto le enseña a entender el significado de las palabras en su contexto.
- Next Sentence Prediction (NSP): Se le dan dos frases y BERT predice si la segunda frase es la que realmente sigue a la primera en el texto original o si es una frase aleatoria. Esto le ayuda a entender la relación entre frases.

2. Ajuste Fino (Fine-Tuning) para Tareas Específicas:

Una vez pre-entrenado, BERT se puede adaptar (fine-tune) para tareas concretas de NLP:

Se añade una capa de salida pequeña y específica para la tarea deseada (ej. una capa de clasificación para análisis de sentimiento, o capas para identificar inicio/fin de respuesta en Question Answering).

Se entrena el modelo completo (BERT + nueva capa) con un conjunto de datos etiquetado y más pequeño, específico para esa tarea. Los pesos de BERT se ajustan ligeramente, aprovechando el conocimiento general ya aprendido.

Este proceso es mucho más rápido y necesita menos datos que entrenar un modelo desde cero.

3. Impacto de los Modelos Preentrenados en NLP:

Mejora del Rendimiento: Han conseguido resultados estado del arte en muchísimas tareas de NLP.

Transferencia de Aprendizaje: El conocimiento aprendido de grandes cuerpos de texto se transfiere eficazmente a tareas con menos datos.

Democratización: Permiten obtener buenos resultados sin necesidad de recursos computacionales masivos para entrenar desde cero.

Nuevo Paradigma: El "pre-entrenamiento y fine-tuning" es ahora el estándar.

4. Ejemplos de Aplicaciones Prácticas:

- Mejora de Motores de Búsqueda: (Ej. Google Search) para entender mejor las consultas.
- Análisis de Sentimiento: Clasificar opiniones de clientes.
- Sistemas de Respuesta a Preguntas (QA): Chatbots y asistentes virtuales.
- Clasificación de Texto: Detección de spam, categorización de noticias.
- Reconocimiento de Entidades Nombradas (NER): Identificar personas, lugares, organizaciones.

5 (1 Punto) Limitaciones y consideraciones éticas

Identifica algunas de las limitaciones de los modelos Transformer en términos de sesgo, consumo de recursos y dependencia de grandes cantidades de datos. Discute las consideraciones éticas asociadas con su uso en aplicaciones reales.

Limitaciones

- **Sesgo (Bias):**

Problema: Los Transformers aprenden de los datos con los que se entrenan. Si estos datos contienen sesgos sociales, culturales o históricos (ej. estereotipos de género, raciales), el modelo los aprenderá y los perpetuará o incluso amplificará en sus predicciones y generaciones.

Impacto: Puede llevar a decisiones injustas o discriminatorias en aplicaciones reales.

- **Consumo de Recursos:**

Problema: Entrenar modelos Transformer grandes (como GPT-3 o T5) requiere una enorme cantidad de poder computacional (GPUs/TPUs) y energía. Esto tiene un coste económico y un impacto ambiental considerable (huella de carbono).

Impacto: Limita el acceso a la investigación y desarrollo a organizaciones con grandes recursos y plantea preocupaciones sobre la sostenibilidad. La inferencia (uso del modelo ya entrenado) también puede ser costosa para aplicaciones a gran escala.

- **Dependencia de Grandes Cantidades de Datos:**

Problema: Para alcanzar su máximo rendimiento, los Transformers necesitan ser pre-entrenados con corpus de texto masivos. No todas las lenguas o dominios específicos disponen de tales cantidades de datos.

Impacto: Puede llevar a un rendimiento inferior en lenguas con pocos recursos o en tareas muy especializadas donde los datos son escasos. Crea una barrera de entrada para la creación de modelos potentes.

Consideraciones Éticas

- **Discriminación y Equidad:**

Como se mencionó, los sesgos en los datos pueden llevar a que los sistemas basados en Transformers discriminen a ciertos grupos en áreas como la contratación, la concesión de créditos o incluso en el sistema judicial.

- **Desinformación y "Deepfakes":**

La capacidad de modelos como GPT para generar texto altamente realista y coherente puede ser explotada para crear noticias falsas, propaganda o suplantar identidades a gran escala, erosionando la confianza y manipulando la opinión pública.

- **Falta de Transparencia:**

A menudo es difícil entender por qué un Transformer toma una decisión específica. Esta naturaleza de "caja negra" complica la rendición de cuentas cuando se cometen errores, especialmente en aplicaciones críticas.

6 (1 Punto) Aplicaciones en Big Data:

Propón un caso de uso de un modelo Transformer en el ámbito de Big Data. Describe cómo utilizarías este modelo para analizar datos masivos, como datos de redes sociales, datos financieros o datos médicos. Explica cómo abordarías los desafíos asociados con este tipo de datos.

Análisis avanzado de la percepción pública y la detección de tendencias a partir de datos masivos generados en redes sociales

Se emplearía un modelo Transformer, como BERT o sus variantes optimizadas (ej. DistilBERT), para procesar grandes volúmenes de texto no estructurado (publicaciones, comentarios) provenientes de diversas plataformas. La ingesta y el procesamiento inicial de estos datos se gestionarían con herramientas de Big Data como Apache Spark y Kafka, capaces de manejar la velocidad y el volumen inherentes.

El modelo Transformer, pre-entrenado en un corpus general y luego ajustado con datos específicos, se utilizaría para:

- Análisis de Sentimiento a Gran Escala: Clasificar el tono (positivo, negativo, neutro) de las menciones relacionadas con una entidad (ej. una marca, un producto, una figura pública), permitiendo una comprensión matizada del sentimiento predominante.
- Identificación Dinámica de Temas y Tendencias: Descubrir y agrupar los principales temas de conversación y las tendencias emergentes en tiempo real, superando las

limitaciones de los métodos tradicionales de modelado de temas gracias a la comprensión contextual del Transformer.

Los desafíos principales, como el volumen de datos, se abordarían mediante el procesamiento distribuido con Spark y la optimización de la inferencia del modelo. La variedad del lenguaje (jerga, errores, multilingüismo) es inherentemente mejor manejada por los Transformers debido a su arquitectura basada en atención y su pre-entrenamiento masivo. La mitigación de sesgos requeriría una cuidadosa selección de datos para el fine-tuning y un monitoreo continuo del rendimiento del modelo.

7 (1 Punto) Optimización y eficiencia

Los modelos Transformer pueden ser costosos en términos de recursos computacionales. Investiga técnicas de optimización y eficiencia para estos modelos, como el uso de Transformers ligeros o técnicas de pruning y quantization. Proporciona ejemplos de cómo estas técnicas pueden mejorar la eficiencia sin sacrificar el rendimiento.

Transformers Ligeros (Lightweight Transformers):

Son arquitecturas de Transformer diseñadas o modificadas para reducir el número de parámetros y operaciones computacionales, buscando mantener un rendimiento competitivo.

Métodos Comunes: Incluyen la factorización de matrices de peso, la compartición de parámetros entre capas (ej. ALBERT), y el desarrollo de mecanismos de atención más eficientes que el estándar $O(n^2)$, como la atención dispersa (ej. Longformer, BigBird) o proyecciones lineales (ej. Linformer), que pueden alcanzar complejidad lineal $O(n)$.

Ejemplo: DistilBERT, una versión destilada de BERT, logra una reducción de tamaño de aproximadamente el 40% y un aumento de velocidad del 60% respecto a BERT-base, conservando cerca del 97% de su capacidad en tareas de comprensión del lenguaje. Esto lo hace adecuado para entornos con recursos limitados.

Pruning:

Es una técnica que consiste en eliminar conexiones o pesos considerados menos significativos dentro del modelo neuronal.

Tipos y Proceso: El Pruning puede ser no estructurado (eliminación de pesos individuales) o estructurado (eliminación de componentes enteros como neuronas o cabezas de atención). Generalmente, implica entrenar un modelo denso, identificar y eliminar los pesos, y opcionalmente realizar un re-entrenamiento (fine-tuning) para recuperar rendimiento.

Impacto: La poda estructurada es particularmente efectiva para reducir el tamaño del modelo y acelerar la inferencia en hardware estándar. Se ha demostrado que es posible eliminar un alto porcentaje de pesos en modelos como BERT con una pérdida de rendimiento mínima tras el fine-tuning.

Quantization (Cuantización):

Es el proceso de reducir la precisión numérica utilizada para representar los pesos y/o las activaciones del modelo, usualmente de punto flotante de alta precisión (ej. FP32) a formatos de menor precisión (ej. FP16, INT8).

Enfoques: Puede realizarse post-entrenamiento (Post-Training Quantization, PTQ), que es más simple, o durante el entrenamiento (Quantization-Aware Training, QAT), que simula la cuantización en el proceso de aprendizaje para mejorar la robustez y minimizar la pérdida de precisión.

Beneficios: La cuantización a INT8, por ejemplo, puede reducir el tamaño del modelo aproximadamente cuatro veces y acelerar significativamente la inferencia (2-4x o más), especialmente en hardware con soporte para operaciones de baja precisión, con una degradación del rendimiento a menudo inferior al 1% si se implementa con QAT.

8 (1 Punto) Construcción de un modelo transformer personalizado:

Dada una tarea específica de procesamiento del lenguaje natural (por ejemplo, traducción automática, resumen de texto, o respuesta a preguntas), diseña un modelo Transformer personalizado. Describe el proceso de diseño, las técnicas de preprocesamiento de datos utilizadas, y las métricas de evaluación que emplearás para medir el rendimiento del modelo.

1. Diseño del Modelo Transformer Personalizado

Arquitectura Base: Encoder-Decoder Transformer estándar.

Encoder: Procesa el artículo de entrada.

Decoder: Genera el resumen token a token, atendiendo a la salida del encoder y a los tokens previamente generados.

Personalización Clave:

- Mecanismo Puntero-Generador (Pointer-Generator): Permite al modelo copiar palabras directamente del texto fuente (útil para nombres, cifras, OOV) o generar desde su vocabulario. Mejora la fidelidad y manejo de OOV.
- Parámetros Ajustables: Número de capas (N_e, N_d), dimensión del modelo (d_model), número de cabezas de atención, etc., se ajustan según los recursos y datos.

2. Preprocesamiento de Datos Clave

Datos: Pares de (artículo, resumen de referencia).

Pasos Esenciales:

Limpieza de Texto: Eliminar ruido (HTML, caracteres especiales).

Tokenización por Subpalabras: Usar SentencePiece o BPE para manejar vocabularios fijos y palabras raras.

Tokens Especiales: Añadir <s>, </s>, [PAD], [SOS], [EOS] según sea necesario para delimitar secuencias y guiar la generación.

Truncamiento/Relleno: Asegurar longitudes de secuencia uniformes para los lotes.

3. Métricas de Evaluación del Rendimiento

Automáticas:

- ROUGE (ROUGE-1, ROUGE-2, ROUGE-L): Mide la superposición de n-gramas y la subcadena común más larga con resúmenes de referencia. Es el estándar.
- BERTScore: Evalúa la similitud semántica contextual entre el resumen generado y el de referencia usando embeddings de BERT.

Humanas:

- Fluidez: Gramaticalidad y legibilidad.
- Coherencia: Conexión lógica de ideas.
- Relevancia/Informatividad: Captura de puntos clave.
- Factualidad: Precisión respecto al original.