Práctica Calificada 1 CC0C2

Fecha de entrega: 28 de septiembre

Puntaje máximo: 20 puntos

Entrega del proyecto: 8 puntos

Exposición del proyecto: 12 puntos

Debes presentar un repositorio donde se encuentre todos tus resultados.

Instrucciones generales:

1. Cada grupo debe estar compuesto por 2 estudiantes.

2. Los proyectos serán asignados por orden de elección, asegurando que cada grupo trabaje en un proyecto diferente.

3. La fecha límite para la entrega del proyecto es el 28 de septiembre. El incumplimiento de la fecha implica una calificación de 0.

4. Las exposiciones tendrán lugar en la misma fecha de entrega. Se asignarán 15 minutos a cada grupo para exponer su proyecto, seguidos de 5 minutos para preguntas.

Proyectos disponibles:

Proyecto 1: Desarrollo de un tokenizador multimodal basado en expresiones regulares y BPE

Descripción: Desarrolla un tokenizador que combine métodos de tokenización top-down (basado en reglas y expresiones regulares) con métodos bottom-up como Byte-Pair Encoding (BPE). El tokenizador deberá ser capaz de manejar tanto palabras completas como subpalabras, dependiendo de la configuración seleccionada. Utiliza expresiones regulares avanzadas, incluyendo:

- Disyunciones
- Agrupaciones y operadores de precedencia
- Sustituciones
- Grupos de captura
- Lookahead Assertions

El sistema debe permitir una tokenización flexible para diferentes lenguajes. Realiza pruebas con un corpus grande como el **Penn Treebank** o el **Corpus Wikipedia**.

Objetivo: Profundizar en el uso avanzado de expresiones regulares y en los principios detrás de la tokenización de subpalabras con BPE. Implementar un tokenizador multimodal que pueda adaptarse a diferentes lenguajes y enfoques de segmentación.

Resultado esperado:

- Un tokenizador funcional que aplique reglas de tokenización top-down y bottom-up.
- Comparación de eficiencia y precisión de las diferentes técnicas de tokenización.
- Documentación que describa la lógica detrás de las expresiones regulares y el algoritmo de BPE.

Proyecto 2: Implementación y comparación de algoritmos de tokenización subpalabra: BPE, WordPiece y SentencePiece

Descripción: Implementa y compara tres técnicas de tokenización subpalabra: **Byte-Pair Encoding (BPE)**, **WordPiece** y **SentencePiece**. Estas técnicas se utilizan ampliamente en tareas de procesamiento del lenguaje natural para manejar vocabularios grandes y palabras raras. Aplica cada algoritmo a un corpus de gran tamaño, como el **Corpus de Noticias Reuters**, y compara su eficiencia, tamaño del vocabulario generado, y calidad en tareas posteriores de procesamiento de texto (como la traducción automática).

Objetivo: Comprender los fundamentos de los diferentes enfoques de tokenización subpalabra, su implementación y su impacto en aplicaciones de NLP.

Resultado esperado:

- Implementaciones funcionales de los tres algoritmos de tokenización.
- Análisis comparativo sobre el tamaño del vocabulario, precisión, y eficiencia.
- Documentación sobre las diferencias técnicas entre BPE, WordPiece y SentencePiece.

Proyecto 3: Desarrollo de un modelo de Lenguaje N-grama escalable con suavizado Kneser-Ney

Descripción: Implementa un modelo de lenguaje basado en n-gramas (hasta 5-gramas) y usa **suavizado Kneser-Ney**, una de las técnicas de suavizado más eficaces para manejar el problema de datos escasos. Utiliza un corpus grande como el **Corpus Wikipedia** y evalúa el rendimiento del modelo en términos de **perplejidad**. Además, se espera que el modelo pueda escalarse para manejar grandes volúmenes de datos sin perder eficiencia.

Objetivo: Explorar la construcción y optimización de modelos n-grama con técnicas de suavizado avanzadas y resolver problemas de escalabilidad para trabajar con grandes corpus.

Resultado esperado:

- Modelo de lenguaje n-grama funcional con suavizado Kneser-Ney.
- Evaluación del rendimiento usando perplejidad.
- Implementación eficiente que pueda manejar corpus de gran escala.

Proyecto 4: Sistema de normalización y segmentación de oraciones utilizando Levenshtein distance y modelos de lenguaje

Descripción: Desarrolla un sistema que realice la normalización de palabras utilizando técnicas como **lemmatización** y **stemming**, seguido de una segmentación de oraciones utilizando un modelo de lenguaje basado en n-gramas. Usa el **algoritmo de Levenshtein** para corregir errores ortográficos y mejorar la calidad del texto antes de segmentar las oraciones. El modelo de lenguaje n-grama deberá ser evaluado en términos de precisión en la segmentación y generación de secuencias válidas de palabras.

Objetivo: Integrar múltiples técnicas de preprocesamiento de texto y generación de secuencias para mejorar la segmentación y corrección de errores en textos no normalizados.

Resultado esperado:

- Un sistema que corrija errores de palabras y segmente oraciones eficientemente.
- Evaluación de la precisión del sistema usando un corpus etiquetado.
- Comparación entre lematización y stemming en términos de calidad de normalización.

Proyecto 5: Generación y evaluación de oraciones mediante modelos de lenguaje N-grama y medición de perplejidad

Descripción: Desarrolla un modelo de lenguaje n-grama que sea capaz de generar oraciones a partir de un corpus grande, como el **Corpus OpenSubtitles**. Utiliza **MLE** para estimar las probabilidades y experimenta con suavizado como **interpolación** y **backoff**. Evalúa el modelo utilizando **perplejidad** y analiza la coherencia de las oraciones generadas. El modelo debe comparar cómo varían las métricas al usar diferentes órdenes de n-gramas (unigramas, bigramas, trigramas, etc.).

Objetivo: Comprender cómo los modelos n-grama generan secuencias de texto y cómo se evalúan utilizando la perplejidad como métrica.

Resultado esperado:

- Un modelo de lenguaje n-grama capaz de generar oraciones.

- Evaluación del modelo en términos de perplejidad y coherencia de las oraciones.
- Análisis comparativo entre diferentes órdenes de n-gramas.

Proyecto 6: Análisis del sobreajuste en modelos de lenguaje N-grama y técnicas de regularización

Descripción: Investiga el fenómeno del sobreajuste en modelos de lenguaje basados en n-gramas entrenados en diferentes corpus, como el **Corpus Europarl** o el **Corpus Brown**. Desarrolla un modelo n-grama que aplique técnicas de regularización, como **suavizado de interpolación** y **backoff**, para evitar el sobreajuste. Evalúa el modelo en un conjunto de prueba y analiza cómo las técnicas de regularización afectan su capacidad de generalización.

Objetivo: Explorar cómo evitar el sobreajuste en modelos n-grama utilizando técnicas de regularización y evaluando su capacidad de generalización en datos no vistos.

Resultado esperado:

- Modelo n-grama entrenado con técnicas de regularización.
- Comparación entre modelos con y sin regularización.
- Análisis sobre el comportamiento del modelo en diferentes corpus.

Proyecto 7: Implementación de un corrector ortográfico y gramatical basado en modelos de lenguaje y distancia de edición

Descripción: Desarrolla un corrector ortográfico que combine el **algoritmo de Levenshtein** para identificar palabras mal escritas y un modelo n-grama para verificar la corrección gramatical de la secuencia de palabras. El corrector debe sugerir la mejor corrección en función de la menor distancia de edición y la probabilidad de la secuencia según el modelo de lenguaje.

Objetivo: Integrar el algoritmo de Levenshtein y los modelos de lenguaje para desarrollar un corrector que maneje tanto errores ortográficos como gramaticales.

Resultado esperado:

- Un corrector ortográfico funcional que maneje tanto errores léxicos como gramaticales.
- Evaluación de la precisión del corrector usando un corpus de pruebas.
- Comparación entre sugerencias basadas en distancia de edición y probabilidad de secuencia.

Proyecto 8: Desarrollo de un tokenizador adaptativo multilingüe utilizando técnicas de aprendizaje automático

- **Descripción**: Implementa un tokenizador adaptable a diferentes lenguajes utilizando **técnicas de aprendizaje automático**. El tokenizador debe ser capaz de aprender reglas específicas de tokenización para cada idioma a partir de un corpus multilingüe como el **Common Crawl** o el **Corpus Tatoeba**. Evalúa cómo el tokenizador se adapta a diferentes lenguas y compara su rendimiento con enfoques tradicionales de tokenización basada en reglas.
- **Objetivo**: Explorar el uso de aprendizaje automático para diseñar un tokenizador que se adapte a las particularidades de múltiples lenguajes.
- **Resultado esperado**:
- Un tokenizador adaptable a diferentes lenguajes mediante técnicas de aprendizaje automático.
- Comparación entre el tokenizador aprendido y enfoques tradicionales.
- Evaluación de la precisión del tokenizador en diferentes lenguas.

Rúbricas de evaluación:

1. Entrega del proyecto (8 puntos):

La entrega debe incluir el código fuente, la documentación del proyecto, y los resultados de las pruebas realizadas con el corpus asignado.

Criterio	Puntos	Descripción
Funcionalidad del código	3	El código debe implementar correctamente el proyecto propuesto y ser completamente funcional, sin errores que afecten su desempeño.
Eficiencia del algoritmo	2	El código debe demostrar eficiencia en el procesamiento, especialmente en proyectos que tratan con grandes volúmenes de datos.
Claridad y estructura del código	1.5	El código debe estar bien estructurado, con comentarios claros y buenas prácticas de programación (modularización, nombres descriptivos, etc.).

Documentación del proyecto	1.5	La documentación debe explicar la implementación, las decisiones técnicas y cómo ejecutar el proyecto correctamente.

2. Exposición del proyecto (12 puntos):

Cada grupo tendrá 15 minutos para exponer su proyecto, seguidos de 5 minutos de preguntas y respuestas.

Criterio	Puntos	Descripción
Claridad en la explicación	4	El grupo debe explicar el proyecto de manera clara, estructurada y coherente, destacando los aspectos clave de su implementación.
Entendimiento técnico	3	El grupo debe demostrar un entendimiento profundo de los conceptos aplicados en el proyecto (ej: tokenización, modelos n-grama, suavizado).
Resultados y análisis	3	El grupo debe presentar los resultados obtenidos de manera clara, con análisis crítico sobre el rendimiento del modelo o algoritmo implementado.
Manejo de preguntas	2	El grupo debe ser capaz de responder a las preguntas de los compañeros o del profesor de manera adecuada y demostrando comprensión del tema.