Relatório LAMIA 24
Prática: Processamento de Linguagem Natural (NLP) (III)

Kaique Medeiros Lima

1 Introdução

O Processamento de Linguagem Natural (PLN) envolve técnicas como tokenização e redes neurais para entender e gerar texto. Este conteúdo explora como essas ferramentas são aplicadas em tarefas como o reconhecimento de sarcasmo e a geração de poesia, utilizando redes neurais recorrentes (RNNs) e LSTMs.

2 Descrição da atividade

2.1 Natural Language Processing - Tokenization (NLP Zero to Hero - Part 1)

O vídeo aborda a tokenização, processo de transformar palavras ou frases em números que um computador pode processar. Em vez de codificar letras individualmente, utiliza-se a codificação de palavras inteiras. Exemplos como "I love my dog" e "I love my cat" são representados numericamente como sequências de tokens. A biblioteca TensorFlow Keras em Python é usada para criar um tokenizer, que atribui números a palavras e permite a manipulação de grandes volumes de texto. O tokenizer lida com exceções, como pontuações, sem criar tokens desnecessários. Esse processo prepara os dados para redes neurais, facilitando a análise ou geração de textos.

2.2 Sequencing - Turning sentences into data (NLP Zero to Hero - Part 2)

Neste vídeo, é discutido o processo de transformação de frases em sequências de números, dando sequência à tokenização das palavras. O método `texts_to_sequences` do tokenizer é utilizado para converter frases em tokens numéricos. Quando palavras novas, que não estão no conjunto de treinamento inicial, são encontradas, isso pode causar problemas, mas é resolvido com a introdução de um token `Out-Of-Vocabulary (OOV)`, que substitui palavras não reconhecidas por um token especial.

Para lidar com o comprimento variável das frases, é aplicado o padding, utilizando a função `pad_sequences` para padronizar o tamanho das sequências, adicionando zeros. Isso garante que todas as sequências tenham o mesmo tamanho, com a flexibilidade de adicionar padding no início ou no final das frases. Além disso, se as frases ultrapassarem um tamanho máximo pré-estabelecido, elas podem ser truncadas. O vídeo apresenta

técnicas para preparar dados de texto para serem alimentados em uma rede neural, com o conteúdo seguinte focado no treinamento de um modelo para classificação de texto.

2.3 Training a model to recognize sentiment in text (NLP Zero to Hero - Part 3)

O vídeo mostra como construir um classificador para identificar sarcasmo em textos, utilizando um conjunto de dados de manchetes rotuladas. O processo começa com o carregamento dos dados em formato JSON, que são convertidos para um formato utilizável em Python. Em seguida, o texto é pré-processado com tokenização e padding, transformando as manchetes em sequências numéricas. O conjunto de dados é dividido em treino e teste, e o modelo de rede neural usa embeddings para representar palavras. Após o treinamento, o classificador atinge 81% de precisão, conseguindo identificar sarcasmo em novas sentenças.

2.4 ML with Recurrent Neural Networks (NLP Zero to Hero - Part 4)

Neste vídeo, o professor explica o conceito de Redes Neurais Recorrentes (RNNs) e sua aplicação na geração de texto. Ele introduz como as RNNs consideram a sequência de dados, contrastando-as com redes neurais tradicionais, onde a ordem dos dados não importa. Ao gerar texto, a ordem das palavras é crucial, e as RNNs são projetadas para lidar com sequências em que o contexto das palavras anteriores é importante para prever a próxima. Ele usa a sequência de Fibonacci como exemplo para explicar o funcionamento das RNNs, onde a saída de cada etapa é alimentada na próxima, criando um ciclo de dados que é processado ao longo do tempo.

Ele também discute a limitação das RNNs básicas, em que palavras distantes em uma frase podem não influenciar efetivamente as previsões devido ao enfraquecimento do contexto conforme ele se espalha. Esse problema pode ser resolvido por um tipo mais avançado de RNN chamado Long Short-Term Memory (LSTM), que retém o contexto por sequências mais longas, melhorando a previsão de palavras que dependem de informações distantes, como no exemplo da frase sobre a Irlanda.

2.5 Long Short-Term Memory for NLP (NLP Zero to Hero - Part 5)

O vídeo explica como as Redes Neurais Recorrentes com Memória de Longo Prazo (LSTM) ajudam a manter o contexto em frases mais longas. Usando o exemplo de uma frase sobre Irlanda, ele mostra como o LSTM pode lembrar palavras anteriores, como "Ireland", para prever corretamente "Gaelic". A arquitetura LSTM mantém um "estado de célula" para preservar o contexto ao longo do tempo. Ele também menciona LSTMs bidirecionais, que analisam o contexto das palavras tanto para frente quanto para trás. O vídeo inclui um exemplo de código para configurar e ajustar camadas LSTM em modelos.

2.6 Training an AI to create poetry (NLP Zero to Hero - Part 6)

O vídeo explica como treinar um modelo para gerar poesia com base nas letras de músicas tradicionais irlandesas, utilizando técnicas de NLP (Processamento de Linguagem Natural). O processo começa com a tokenização das letras das músicas, seguida pela criação de sequências e n-grams para prever a próxima palavra. O modelo é treinado com uma rede neural simples, usando uma camada de embedding e uma LSTM bidirecional. Após o treinamento, o modelo pode gerar texto, completando sequências de palavras. O resultado final é uma geração de texto com cerca de 70% de precisão, permitindo a criação de novas poesias a partir de uma sequência inicial de palavras.

2.7 Natural Language Processing with spaCy and Python - Course for Beginners

O Processamento de Linguagem Natural (PLN) visa permitir que as máquinas compreendam a linguagem humana. Durante o curso, foi utilizado a biblioteca spaCy, que oferece uma estrutura robusta para lidar com textos. Essa biblioteca armazena informações por palavras em vez de caracteres, o que amplia as possibilidades de processamento. Através de funções como `.ents` para separar frases e métodos como `left_edge`, `ent_type` e `lemma_`, é possível realizar diversas análises detalhadas no texto. Além disso, a ferramenta `displacy` facilita a visualização de entidades destacadas no texto, enquanto o `ruler` permite ajustar erros encontrados nas entidades por meio de padrões. O `matcher` é usado para buscar padrões específicos sem modificar o conteúdo, e o `language` ajuda a filtrar informações irrelevantes.

3 Conclusão

As técnicas de PLN, como tokenização e LSTMs, são fundamentais para processar e gerar texto de forma inteligente. Elas permitem o desenvolvimento de modelos capazes de realizar tarefas como análise de sentimentos e criação de textos, mostrando o potencial do PLN em diversas aplicações.