Exercício 3

Suponha que você tenha uma tarefa específica de processamento de linguagem natural (PLN) para a qual deseja criar um modelo de alto desempenho. Você tem acesso a um modelo de linguagem pré-treinado e deseja realizar o fine-tuning desse modelo para sua tarefa específica.

1. Escolha uma tarefa de NLP específica, de sua preferência, para a qual você deseja adaptar o modelo de linguagem, como classificação de sentimentos, resumo de texto, geração de texto ou qualquer outra tarefa.

Vamos escolher a tarefa de **classificação de sentimentos**. Especificamente, a tarefa envolve a classificação de textos (como resenhas de produtos ou posts em redes sociais) em categorias como "positivo", "negativo" e "neutro".

1. Como você faria a coleta e processamento desses dados?

**Coleta de Dados**:

* + **Fontes de Dados**: Utilizar datasets já disponíveis como o **IMDb**, que contém resenhas de filmes rotuladas, ou o **Twitter Sentiment Analysis Dataset**, que contém tweets rotulados.
  + **Web Scraping**: Se necessário, podemos realizar web scraping em sites de avaliações de produtos ou redes sociais para coletar mais dados.
  + **APIs**: Usar APIs, como a API do Twitter, para coletar tweets com hashtags específicas e realizar a rotulagem manual ou semiautomática.

**Processamento de Dados**:

* + **Limpeza de Texto**: Remover URLs, menções, emojis e outros caracteres especiais desnecessários.
  + **Tokenização**: Dividir o texto em tokens (palavras ou subpalavras).
  + **Remoção de Stop Words**: Eliminar palavras comuns que não agregam valor ao significado (opcional dependendo do modelo pré-treinado).
  + **Lematização/Stemming**: Reduzir as palavras às suas formas base (opcional).
  + **Balanceamento de Dados**: Se as classes estiverem desequilibradas, técnicas como oversampling, undersampling ou data augmentation podem ser aplicadas.

1. Descreva o processo de fine-tuning do modelo de linguagem pré-treinado para a tarefa específica.

**Escolha do Modelo Pré-Treinado**:

* + Usaremos um modelo pré-treinado como o **BERT** (Bidirectional Encoder Representations from Transformers), que já foi treinado em uma grande quantidade de dados textuais.

*OBS: Devido à limitação de processamento na minha versão do Colab, no código, usei apenas* ***5% dos dados do IMDb*** *e o* ***DistilBERT*** *que é um modelo mais leve que o BERT padrão.*

**Preparação do Dataset**:

* + **Divisão de Dados**: Dividir o dataset em conjuntos de treinamento, validação e teste.
  + **Codificação das Etiquetas**: As etiquetas (positivo, negativo, neutro) serão codificadas em números (Ex: 0, 1, 2).

**Configuração do Fine-Tuning**:

* + **Adaptar a Camada de Saída**: A última camada do modelo BERT é adaptada para ter uma saída com o número de classes na tarefa de classificação de sentimentos.
  + **Configuração de Hiperparâmetros**: Definir a taxa de aprendizado, número de épocas, tamanho do batch, etc.
  + **Treinamento**: Treinar o modelo com o dataset específico, ajustando os pesos do modelo para a tarefa de classificação de sentimentos.
  + **Validação**: Durante o treinamento, usar o conjunto de validação para ajustar os hiperparâmetros e evitar overfitting.

1. Como você avaliaria o desempenho do modelo fine-tuned, especifique as métricas.

**Métricas de Avaliação**:

* + **Acurácia**: Proporção de previsões corretas sobre o total de previsões.
  + **Precisão, Recall e F1-Score**: Especialmente úteis em cenários com classes desbalanceadas. A precisão mede a quantidade de verdadeiros positivos em relação aos falsos positivos, enquanto o recall mede a quantidade de verdadeiros positivos em relação aos falsos negativos. O F1-score é a média harmônica entre precisão e recall.
  + **Matriz de Confusão**: Visualização da performance do modelo em termos de verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos.

**Teste em Conjunto de Dados Desconhecido**:

* + Utilizar o conjunto de teste para avaliar a generalização do modelo fine-tuned.

1. Descreva como você implantaria o modelo fine-tuned em um ambiente de produção para realizar previsões em novos dados.

**Salvamento do Modelo**:

* + Após o treinamento, salvar o modelo treinado em um formato como .h5 ou .pt, dependendo do framework (TensorFlow ou PyTorch).

**Criação de um Serviço de API**:

* + Desenvolver uma API RESTful usando frameworks como Flask ou FastAPI, onde o modelo pode receber dados de entrada (texto) e retornar a previsão de sentimento.
  + Implementar logging para monitorar o uso do modelo em produção.

**Escalabilidade**:

* + Se necessário, implementar mecanismos de escalabilidade, como Kubernetes, para lidar com altas demandas de requisições.
  + Utilizar balanceamento de carga e caching para melhorar a eficiência.

**Monitoramento Contínuo**:

* + Implementar monitoramento para detectar qualquer degradação de desempenho do modelo ao longo do tempo (data drift).
  + Atualizar periodicamente o modelo com novos dados coletados em produção.

EXTRA - você pode explorar o ajuste de hiperparâmetros do modelo fine-tuned, como o tamanho da rede, a taxa de aprendizado e o número de épocas de treinamento, para otimizar o desempenho.

**Ajuste de Hiperparâmetros**

1. **GridSearchCV ou RandomSearchCV**:
   * Implementar técnicas como GridSearchCV ou RandomSearchCV para encontrar a melhor combinação de hiperparâmetros como taxa de aprendizado, número de épocas, tamanho do batch, entre outros.
   * Durante o ajuste de hiperparâmetros, é crucial manter o balanceamento entre overfitting e underfitting.
2. **Taxa de Aprendizado**:
   * Experimentar diferentes taxas de aprendizado para garantir que o modelo esteja convergindo adequadamente durante o treinamento.
3. **Número de Épocas**:
   * Avaliar o número de épocas ideal para garantir que o modelo não esteja subtreinado ou supertreinado. Isso pode ser monitorado através da curva de perda e da performance no conjunto de validação.
4. **Tamanho do Batch**:
   * Ajustar o tamanho do batch pode influenciar diretamente a performance do modelo. Tamanhos maiores podem acelerar o treinamento, enquanto tamanhos menores podem oferecer mais estabilidade.