TPF-02: Reprodução do Artigo *"DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter"*

Henrique Alves
Disciplina de Redes Neurais
Universidade Federal de Viçosa – Campus Florestal
Florestal, MG
henrique.a.campos@ufv.br

1 Introdução

Modelos de linguagem baseados em Transformers, como o BERT, apresentam elevado custo computacional e de memória, tanto para treinamento quanto para inferência. O artigo *"DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter"* [1] aborda esse problema de forma direta, propondo uma versão mais compacta e eficiente do BERT.

O DistilBERT reduz em aproximadamente 40% o número de parâmetros e é cerca de 60% mais rápido, mantendo mais de 97% da performance original. Essa eficiência é alcançada por meio do método de *knowledge distillation*, onde um modelo aluno aprende a reproduzir o comportamento interno e as previsões de um modelo professor.

O objetivo deste relatório é documentar a reprodução do experimento de *fine-tuning* do DistilBERT na tarefa de análise de sentimentos **SST-2** Stanford Sentiment Treebank do benchmark GLUE, além de discutir os desafios técnicos, os resultados e as diferenças observadas em relação ao artigo original.

2 Metodologia

2.1 Ambiente e Ferramentas

O experimento foi conduzido no *Google Colab* [5], ambiente em nuvem que oferece suporte gratuito a GPU. No entanto, durante a execução, foi detectada a ausência de acelerador, resultando em um treinamento realizado totalmente em CPU.

Para garantir reprodutibilidade e evitar conflitos de versão, foi utilizada a seguinte configuração de ambiente:

- `Python 3.12`;
- `torch` 2.4.0 [4];
- `transformers` 4.44.2 [2];
- `datasets` 2.20.0 [3];
- `evaluate` 0.4.2;
- `numpy` 1.26.4;
- `pandas` 2.2.2;
- `scikit-learn` 1.5.1.

Essas versões foram instaladas manualmente com o comando:

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

```
!pip install -q numpy==1.26.4 pandas==2.2.2 pyarrow==15.0.2 \
datasets==2.20.0 transformers==4.44.2 evaluate==0.4.2 scikit-learn==1.5.1
```

Essa configuração seguiu as recomendações oficiais das bibliotecas da Hugging Face [2, 3], garantindo compatibilidade com o modelo DistilBERT.

2.2 Preparação dos Dados

Foi utilizado o conjunto `glue/sst2`, composto por 67.349 exemplos de treino, 872 de validação e 1.821 de teste. Cada amostra contém uma sentença e um rótulo binário (0 para sentimento negativo e 1 para positivo).

O pré-processamento seguiu as boas práticas da documentação da Hugging Face [2] e envolveu:

1. **Tokenização:** conversão do texto em *tokens* utilizando o tokenizador do modelo `distilbert-base-uncased`;
2. **Codificação:** transformação das sentenças em `input_ids` e `attention_masks`;
3. **Agrupamento:** uso do `DataCollatorWithPadding` para ajustar dinamicamente o tamanho dos *batches*.

2.3 Configuração e Treinamento

O modelo foi carregado via `AutoModelForSequenceClassification` com duas classes (`num_labels=2`). Os hiperparâmetros utilizados foram:

- **Learning rate:** 2e-5;
- **Épocas:** 3;
- **Batch size:** 16;
- **Weight decay:** 0.01.

O treinamento foi controlado pela classe `Trainer` [2], com estratégia de avaliação por época e salvamento do melhor modelo ao final. O processo completo durou cerca de 11 horas em CPU, com múltiplos avisos do tipo:

```
UserWarning: 'pin_memory' argument is set as true but no
accelerator is found...
```

indicando ausência de GPU.

3 Resultados e Discussão

3.1 Métrica de Avaliação

A métrica principal para a tarefa SST-2 é a *accuracy*, representando a proporção de previsões corretas no conjunto de validação [8].

3.2 Resultados Obtidos

Tabela 1: Log de treinamento e validação por época.

Época	Training Loss	Validation Loss	Accuracy
1	0.1711	0.3103	0.9037
2	0.1223	0.3715	0.9048
3	0.0787	0.4245	0.9071

A melhor performance foi alcançada na **Época 1**, com acurácia de 90.37% e menor perda de validação 0.3103. A execução final reportou:

```
{'eval_loss': 0.3103, 'eval_accuracy': 0.9037}
```

3.3 Análise e Comparação

O artigo original reporta uma acurácia de 91,3% no mesmo conjunto de dados. A diferença inferior a 1% em relação ao valor obtido nesta reprodução indica uma execução bem-sucedida, dentro das variações esperadas em experimentos de aprendizado profundo [4].

Observa-se que a *validation loss* atingiu seu menor valor na primeira época e aumentou nas seguintes, indicando leve *overfitting*. Assim, o melhor modelo corresponde ao ponto em que a perda de validação foi mínima, preservando a capacidade de generalização.

4 Desafios e Aprendizados

4.1 Conflitos de Ambiente e Versão

Durante o experimento, ocorreram erros do tipo `TypeError` ao criar os `TrainingArguments`, devido a incompatibilidades entre versões das bibliotecas. A solução envolveu reinstalar manualmente as versões adequadas e reiniciar o *kernel* do Colab. Casos semelhantes são amplamente discutidos na comunidade da Hugging Face e em fóruns técnicos [6, 7], evidenciando a importância da gestão de dependências em ambientes dinâmicos como o Colab.

4.2 Limitações de Hardware

O treinamento em CPU prolongou o tempo total para aproximadamente 11 horas. Em ambiente com GPU, o mesmo processo levaria cerca de 10 a 15 minutos [5]. Essa diferença destaca a relevância de verificar o hardware disponível antes de iniciar experimentos de *fine-tuning*.

5 Conclusão

A reprodução do artigo *DistilBERT* foi bem-sucedida, confirmando a eficiência do modelo em manter alta acurácia mesmo com significativa redução de tamanho e custo computacional.

Os resultados foram consistentes com os do trabalho original, e a experiência proporcionou aprendizado prático sobre:

- Configuração de ambientes reprodutíveis no Colab;
- Ajuste de hiperparâmetros em modelos Transformer;
- Interpretação de logs de treinamento e análise de *overfitting*;
- Impacto direto da disponibilidade de GPU em tarefas de *deep learning*.

O projeto demonstrou domínio dos conceitos de *fine-tuning* e do ecossistema `HuggingFace`, além de reforçar a importância da experimentação controlada para validação científica.

Referências

[1] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. arXiv preprint arXiv:1910.01108. Disponível em: `https://arxiv.org/abs/1910.01108`

[2] Hugging Face. *Transformers Documentation*. Disponível em: `https://huggingface.co/docs/transformers`. Acesso em: 11 nov. 2025.

[3] Hugging Face. *Datasets Documentation*. Disponível em: `https://huggingface.co/docs/datasets`. Acesso em: 11 nov. 2025.

[4] PyTorch. *PyTorch Documentation*. Disponível em: `https://pytorch.org/docs/stable/index.html`. Acesso em: 11 nov. 2025.

[5] Google Research. *Colaboratory FAQ – Managing Environments and Libraries*. Disponível em: `https://research.google.com/colaboratory/faq.html`. Acesso em: 11 nov. 2025.

[**6**] Hugging Face. *Transformers GitHub Issues*. Disponível em: `https://github.com/huggingface/transformers/issues`. Acesso em: 11 nov. 2025.

[**7**] Stack Overflow. *Hugging Face Transformers – Common Issues and Version Conflicts*. Disponível em: `https://stackoverflow.com/questions/tagged/huggingface-transformers`. Acesso em: 11 nov. 2025.

[**8**] Scikit-learn. *scikit-learn Documentation*. Disponível em: `https://scikit-learn.org/stable/documentation.html`. Acesso em: 11 nov. 2025.