Universidade do Minho
Escola de Engenharia
Mestrado em Engenharia Informática

Aprendizagem Profunda

Ano lectivo 2023/2024

Relatório do trabalho prático

Grupo 8

Miguel Silva Pinto, PG54105

Orlando José da Cunha Palmeira, PG54123

Pedro Miguel Castilho Martins, PG54146

Braga, 31 de Maio de 2024

AP

Índice

1 Introdução

Este relatório foi elaborado no âmbito do projeto prático da unidade curricular de Aprendizagem Profunda e tem como objetivo explorar o desenvolvimento de modelos de *deep learning* utilizando as técnicas abordadas ao longo do semestre.

O nosso grupo ficou com o tema de Tradução Automática, que é um dos problemas clássicos da inteligência artificial. A utilização de técnicas de aprendizagem profunda tem permitido a melhoria na precisão e a naturalidade das traduções entre diversas linguagens.

Todo o trabalho realizado para este projeto pode ser encontrado em orlandopalmeira/Trabalho_AP_2023_2024.

1.1 Objetivos

Os principais objetivos para este projeto prático são os seguintes:

- Explorar o estado da arte do *deep learning* na tradução automática
- Explorar um *dataset* de tradução automática
- Avaliar e comparar modelos já existentes de tradução automática
- Fazer *finetune* a esses modelos de forma a melhorar as suas traduções
- Desenvolver uma aplicação que permita a utilização dos modelos

2 Metodologia

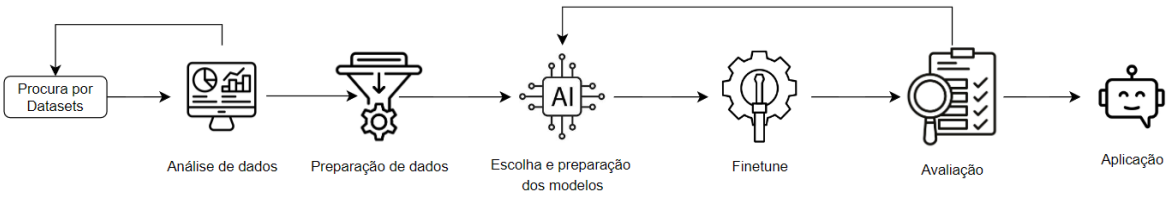


Figura 2.1: Metodologia

Para este projeto a metodologia utilizada foi a que está presente na figura 2.1. Esta metodologia foi desenvolvida para corresponder com os objetivos definidos anteriormente e descrever as diversas etapas do desenvolvimento do nosso projeto.

Assim sendo, apresentamos de forma mais detalhada nossa metodologia que é composta por:

- **Procura por Datasets**: Esta é a primeira etapa e consiste na procura de *datasets* de tradução multilingue capazes de influenciar modelos de tradução já existentes de forma a que estes sejam capazes de fazer traduções mais precisas sobre certos tópicos.
- **Análise de dados**: Nesta etapa é feita uma análise sobre o *dataset* escolhido, onde percebemos qual o seu propósito e se é adequado para o nosso projeto. Este processo ocorreu sobre diversos *datasets* de tradução.
- **Preparação dos dados**: Esta etapa serve para preparar o *dataset* escolhido nas etapas anteriores de forma a que seja utilizado nos modelos de aprendizagem. Foi feita uma extração das línguas que pretendíamos do *dataset* e uma divisão entre dados de treino, avaliação e teste.

- **Escolha e preparação dos modelos**: Nesta etapa o objetivo é encontrar modelos de tradução automática que pudessem ser treinados de forma a produzir traduções mais precisas sobre um certo tópico.
- **Finetune**: Esta etapa baseou-se no *finetune* dos modelos com vários hiperparâmetros utilizando o *dataset* escolhido anteriormente.
- **Avaliação**: Nesta etapa de avaliação utilizamos os modelos *finetuned* com o nosso *dataset* e calculamos a precisão das traduções. Como métrica de avaliação utilizamos o BLEU (Bilingual Evaluation Understudy) por ser uma métrica capaz de avaliar uma tradução feita por um modelo e a tradução esperada, segundo o nosso dataset.
- **Aplicação**: Nesta ultima fase criamos uma aplicação capaz de receber *input* e devolver a tradução esperada em outra língua.

3 Descrição e exploração dos dados

O *dataset* selecionado para o nosso projeto foi um excerto do *Massive* desenvolvido pelo departamento de pesquisa e desenvolvimento da *Amazon* [1].

Este *dataset* foi projetado para ser utilizado no *finetuning* de modelos de tradução automática já existentes, com o objetivo principal de melhorar as suas capacidades em traduzir frases e comandos comuns entre várias línguas com precisão. Com este *dataset* é possível melhorar a compreensão de contexto, a consistência de tradução e o reconhecimento de padrões nos modelos de tradução automática.

O intuito deste *dataset* é aprimorar os modelos de *machine learning* em tarefas de entendimento de linguagem natural (NLU) como:

- **Classificação de intenções**: Esta tarefa baseia-se na compreensão das intenções que a frase transmite. Por exemplo a frase "qual é o tempo esta semana" transmite a intenção de saber o estado meteorológico para esta semana.
- **Anotação de slots**: Esta tarefa ajuda os modelos a detetar informações essenciais à frase. Por exemplo a frase "acorda-me às nove da manhã na sexta-feira" possui 2 *slots*, "nove da manhã" e "sexta-feira" que representam informação especifica sobre a frase que os modelos necessitam de reconhecer.

Este *dataset* é útil para modelos que precisam traduzir comandos dados a um assistente virtual, garantindo que a intenção do cliente e as informações relevantes sejam mantidas durante a tradução.

O *dataset* é composto por 12 colunas, sendo que 10 colunas são frases escritas nas 10 línguas presentes no *dataset*, e as outras 2 colunas são o *id* e o *split* do *dataset* (*train*, *validation*, *test*).

As línguas presentes no *dataset* são as seguintes, inglês, alemão, francês, espanhol, português, hindi, italiano, árabe, holandês e japonês e o número de linhas presentes no *dataset* são (*train*, 11514), (*validation*, 2033), (*test*, 2974).

Para este projeto decidimos explorar apenas 5 das 10 línguas presentes no *dataset* sendo elas o inglês, alemão, francês, espanhol e português. Desta maneira conseguimos com que o treino dos nossos modelos ocorra num tempo razoável e seja mais fácil comprovar a capacidade dos nossos modelos.

4 Descrição dos modelos

Nesta secção vamos apresentar os modelos de tradução automática que exploramos para fazer *finetune* com o *dataset* escolhido. A pesquisa pelos *datasets* foi realizada maioritariamente no *website* Hugging

Face por possuir uma vasta quantidade de modelos de *machine learning* e *datasets*.

Os modelos escolhidos apresentam diferentes capacidades de tradução sendo elas as seguintes:

- O primeiro modelo é capaz de traduzir textos entre Inglês e Português
- O segundo modelo é capaz de traduzir textos entre Inglês e várias línguas, sendo que neste projeto apenas exploramos o alemão, francês, espanhol e português.
- O terceiro modelo é capaz de traduzir textos entre múltiplas línguas para múltiplas línguas.

4.1 Modelo opus-mt-tc-big-en-pt

Este modelo foi desenvolvido pela universidade da Helsínquia e faz parte do projeto OPUS-MT project que tem como objetivo a criação de modelos de tradução entre várias línguas do mundo. O modelo "opus-mt-tc-big-en-pt" [6][5] é um modelo capaz de traduzir textos escritos em língua inglesa para português e possui as seguintes características:

- Arquitetura do modelo: Transformer
- Tamanho do modelo: 233 Milhões de parâmetros

Para utilizarmos este modelo e fazer o seu *finetune*, recorremos à biblioteca Transformers do Hugging Face que permite a importação de modelos. Esta biblioteca também disponibiliza objetos de treino e avaliação dos modelos que iremos utilizar.

Inicialmente fizemos o *import* do modelo juntamente com o seu *tokenizer* para que seja feita a transformação das frases em *tokens* para o modelo processar.

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM

model_name = "Helsinki-NLP/opus-mt-tc-big-en-pt"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
```

De seguida preparamos o *dataset* para o *finetune* do modelo. Esta preparação implicou a adição de um *token* "»por«" no inicio de cada frase de *input* presente no *dataset*. Este *token* indica ao modelo a tarefa que deve realizar, sendo que neste caso a tarefa é traduzir o texto de *input* para português.

A tokenização do *dataset* foi feito da seguinte forma:

```
def tokenize_function_en_pt(example):
    inputs = [f">>por<< {ex}" for ex in example['en_US']]
    targets = example['pt_PT']

    return tokenizer(
        inputs,
        text_target=targets,
        truncation=True, max_length=128
    )

class Dataset():
    def __init__(self):
        train_dataset = load_dataset("Amani27/massive_translation_dataset", split="train")
        valid_dataset = load_dataset("Amani27/massive_translation_dataset",
            split="validation")
        test_dataset = load_dataset("Amani27/massive_translation_dataset", split="test")

        self.tokenized_train_dataset = train_dataset.map(tokenize_function_en_pt,
            batched=True, remove_columns=train_dataset.column_names)
        self.tokenized_valid_dataset = valid_dataset.map(tokenize_function_en_pt,
            batched=True, remove_columns=valid_dataset.column_names)
```

```
        self.tokenized_test_dataset = test_dataset.map(tokenize_function_en_pt,
            batched=True, remove_columns=test_dataset.column_names)
```

Com o *dataset* tokenizado e dividido em 3 *splits* passamos para a etapa de *finetunning*.

4.1.1 Finetune

Para fazer o *finetune* dos modelos recorremos novamente à biblioteca *transformers* utilizamos as seguintes classes:

```
from transformers import Seq2SeqTrainingArguments, Seq2SeqTrainer, DataCollatorForSeq2Seq
```

A classe *Seq2SeqTrainingArguments* permite definir os hiperparâmetros que serão utilizados durante o finetune dos modelos. Permite a alteração do *learning_rate*, *weight_decay* e o número de *epochs* do *finetune*. Também disponibiliza outras opções que podem otimizar o processso de *finetune*.

A classe *Seq2SeqTrainer* permite a criação de objeto *Trainer* que é o objeto que contém o modelo e os dados de treino. Este objeto possui um método *train* onde aplica os argumentos passado e os dados ao modelo para que ele seja retreinado, e possui um método *evaluate* que avalia o modelo com os dados e avaliação que lhe foram passados.

A classe *DataCollatorForSeq2Seq* serve para a criação de um componente no processo de treino que trata do *padding*, *truncation* e prepara o *input* e *output* para que sejam compatíveis com o modelo. Esta preparação feita pelo *DataCollator* permite um treino mais eficiente.

Aqui está um exemplo da preparação feita para o *finetune* deste modelo:

```
data_collator = DataCollatorForSeq2Seq(tokenizer, model=model)

args = Seq2SeqTrainingArguments(
    f"model_opus-mt-tc-big-en-pt-finetuned",
    learning_rate=2e-5,
    weight_decay=0.01,
    num_train_epochs=3,
    per_device_train_batch_size=32,
    per_device_eval_batch_size=32,
    evaluation_strategy="no",
    save_total_limit=0,
    predict_with_generate=True,
    fp16=True,
)

trainer = Seq2SeqTrainer(
    model,
    args,
    train_dataset=dataset.tokenized_train_dataset,
    eval_dataset=dataset.tokenized_valid_dataset,
    data_collator=data_collator,
    tokenizer=tokenizer,
    compute_metrics=compute_metrics,
)
```

4.1.2 Avaliação dos resultados

A avaliação do modelo foi feita a partir de uma função que calculava o valor do BLEU Score do modelo.

```
import evaluate
```

```
2
3  metric = evaluate.load("sacrebleu")
4  def compute_metrics(eval_preds):
5      preds, labels = eval_preds
6      if isinstance(preds, tuple):
7          preds = preds[0]
8
9      decoded_preds = tokenizer.batch_decode(preds, skip_special_tokens=True)
10
11     labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
12     decoded_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)
13
14     decoded_preds = [pred.strip() for pred in decoded_preds]
15     decoded_labels = [[label.strip()] for label in decoded_labels]
16
17     result = metric.compute(predictions=decoded_preds, references=decoded_labels)
18     return {"bleu": result["score"]}
```

Primeiramente utilizamos esta função para calcular a performance do modelo base importado do Hugging Face e obtivemos um BLEU Score de 27.5.

Depois calculamos o BLEU Score após o treino do modelo com vários hyperparâmetros e estes foram os resultados obtidos.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.001	0.1	3	15.7
0.0001	0.01	3	45.6
0.00001	0.01	3	45.9
0.000001	0.01	3	39.9
0.00002	0.01	3	46.5
0.00003	0.01	3	47.1

Tabela 4.1: Resultados obtidos

Os melhores resultados foram obtidos com o Learning_rate a 0.00003 e o Weight_decay a 0.01 com um BLEU Score de 47.1.

4.2 Modelo opus-mt-en-mul

Este modelo semelhante ao anterior também foi desenvolvido pela universidade da Helsínquia e fez parte do mesmo projeto de criação de modelos de tradução automática. O modelo "opus-mt-en-mul" [4] é um modelo capaz de traduzir textos escritos em língua inglesa para múltiplas línguas e possui as seguintes características:

- Arquitetura do modelo: Transformer
- Tamanho do modelo: 77 Milhões de parâmetros

A preparação deste modelo foi bastante semelhante ao anterior, apenas precisando de tratar dos dados que passamos ao modelo de forma diferente. Para isso fizemos o seguinte:

```
1  def tokenize_function(example):
2      inputs = [f">>por<< {ex}" for ex in example['en_US']]
3      inputs += [f">>fra<< {ex}" for ex in example['en_US']]
4      inputs += [f">>spa<< {ex}" for ex in example['en_US']]
5      inputs += [f">>deu<< {ex}" for ex in example['en_US']]
```

```
    targets = example['pt_PT']
    targets += example['fr_FR']
    targets += example['es_ES']
    targets += example['de_DE']

    return tokenizer(
        inputs,
        text_target=targets,
        truncation=True, max_length=128
    )
```

Desta forma o modelo aprende a fazer a tradução entre o inglês e as outras 4 línguas.

4.2.1 Avaliação dos resultados

O processo de finetune foi semelhante ao modelo anterior. No entanto, desta vez calculamos a qualidade da tradução entre as diversas línguas. Estes foram os resultados obtidos:

BLEU Score base do modelo para a tradução entre Inglês e Português é de 20.2.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.00003	0.1	3	39.0
0.00003	0.01	3	39.0
0.000003	0.1	3	29.5
0.0003	0.1	3	41.0

Tabela 4.2: Resultados obtidos (Inglês → Português)

BLEU Score base do modelo para a tradução entre Inglês e Francês é de 24.4.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.0003	0.1	3	50.0
0.00003	0.01	3	45.4
0.000003	0.1	3	34.8

Tabela 4.3: Resultados obtidos (Inglês → Francês)

BLEU Score base do modelo para a tradução entre Inglês e Espanhol é de 25.7.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.0003	0.1	3	47.6
0.00003	0.01	3	46.2
0.000003	0.1	3	37.0

Tabela 4.4: Resultados obtidos (Inglês → Espanhol)

BLEU Score base do modelo para a tradução entre Inglês e Alemão é de 6.0.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.0003	0.1	3	36.3
0.00003	0.01	3	33.2
0.000003	0.1	3	25.2

Tabela 4.5: Resultados obtidos (Inglês → Alemão)

BLEU Score base do modelo para a tradução entre Inglês e todas as línguas é de 19.7.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.0003	0.1	3	44.5
0.0003	0.01	3	44.1

Tabela 4.6: Resultados obtidos (Inglês → Todas as línguas)

4.3 Modelo small100

Este modelo é um modelo de tradução multilingue que ao contrário dos outros é capaz de receber como input várias línguas e traduzir para várias línguas. O modelo "small100" [2][3] possui as seguintes características:

- Arquitetura do modelo: Seq-2-Seq (Encoder-Decoder)
- Tamanho do modelo: 333 Milhões de parâmetros

A preparação dos dados foi semelhante ao modelo anterior com apenas uma pequena mudança no token de inicialização da tarefa que desta vez é necessário indicar-lo da seguinte forma:

```
tokenizer.tgt_lang = "pt" # id da línguagem
```

4.3.1 Avaliação dos resultados

A avaliação deste modelo foi realizada entre Inglês e as outras línguas que estudamos. Desta forma conseguimos fazer uma comparação entre este modelo e o modelo "opus-mt-en-mul".

BLEU Score base do modelo para a tradução entre Inglês e Português é de 0.7.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.001	0.1	3	35.9
0.0001	0.1	3	43.3
0.00001	0.1	3	40.7
0.0001	0.01	3	40.6

Tabela 4.7: Resultados obtidos (Inglês → Português)

BLEU Score base do modelo para a tradução entre Inglês e Francês é de 1.3.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.001	0.1	3	47.0
0.0001	0.1	3	55.0
0.00001	0.1	3	51.5
0.0001	0.01	3	55.3

Tabela 4.8: Resultados obtidos (Inglês → Francês)

BLEU Score base do modelo para a tradução entre Inglês e Espanhol é de 0.8.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.001	0.1	3	25.4
0.0001	0.1	3	51.2
0.00001	0.1	3	48.9
0.0001	0.01	3	50.6

Tabela 4.9: Resultados obtidos (Inglês → Espanhol)

BLEU Score base do modelo para a tradução entre Inglês e Alemão é de 1.6.

Learning_rate	Weight_decay	Num_train_epochs	BLEU Score
0.001	0.1	3	34.3
0.0001	0.1	3	39.4
0.00001	0.1	3	36.3
0.0001	0.01	3	39.7

Tabela 4.10: Resultados obtidos (Inglês → Alemão)

Como podemos ver pelos resultados obtidos este modelo apesar de ter um BLEU Score bastante baixo inicialmente, após o finetune ele foi capaz de melhorar bastante a sua capacidade de compreensão de linguagem natural e de tradução sobre o nosso *dataset*.

4.4 Comparação entre os modelos

Nesta secção vamos fazer uma comparação entre os modelos multilingue estudados.

Linguagen	BLEU Score opus-mt-en-mul	BLEU Score small100
Inglês -> Português	41.0	43.3
Inglês -> Francês	50.0	55.3
Inglês -> Espanhol	47.6	51.2
Inglês -> Alemão	36.3	39.7

Tabela 4.11: Comparação entre os modelos multilingue

Nesta tabela conseguimos perceber que o modelo "small100" obteve melhores resultados sobre todos os pares de línguas comparativamente ao modelo "opus-mt-en-mul". Isto pode indicar que o modelo "small100" é mais influenciável pelos dados de treino que recebe e é melhor quando se pretende que seja utilizado em contextos mais específicos.

5 Aplicação

Para finalizar o nosso projeto decidimos desenvolver uma aplicação que permitisse a utilização dos modelos que exploramos. Os modelos *finetuned* desenvolvidos podem ser encontrados nos seguintes links:

- Miguelcj1/opus-mt-tc-big-en-pt-finetuned
- Miguelcj1/opus-mt-en-mul-finetuned
- Miguelcj1/small100-finetuned

A aplicação foi desenvolvida com recurso à *framework Gradio* que permite a criação de *chatbots* interativos com os quais podemos testar os nossos modelos. Nos anexos estão alguns exemplos da nossa aplicação em funcionamento [A].

6 Conclusão e Trabalho Futuro

Com a conclusão deste trabalho, gostaríamos de refletir sobre o trabalho realizado. Este trabalho focou-se no desenvolvimento e avaliação de modelos de tradução automática. Para tal, exploramos vários modelos de tradução automática realizando *finetuning* com o objetivo de melhorar o seu desempenho em tarefas específicas de tradução.

Consideramos que os resultados que obtivemos com o nosso trabalho demonstram melhorias significativas na compreensão de linguagem natural dos modelos e além disso consideramos a aplicação prática dos modelos através da *framework Gradio* uma boa forma de interação com o trabalho desenvolvido.

Para trabalho futuro consideramos que temos alguns pontos que devemos melhorar ou expandir tal como:

- **Expansão nas linguagens**: Explorar a inclusão de mais linguagens na tradução dos modelos multilingue.
- **Interação Humano-Máquina**: Desenvolver uma interface que permita o uso de voz para que a utilização dos modelos seja mais fácil.
- **Avaliação Qualitativa**: Complementar a métrica de avaliação quantitativa (BLEU), com avaliações qualitativas e *feedback* de utilizadores para entender melhor as limitações e pontos fortes dos modelos.

Em suma, consideramos que o trabalho foi realizado com sucesso e os objetivos que nos propusemos a cumprir foram realizados com sucesso.

Referências

[1] FitzGerald, J., Hench, C., Peris, C., Mackie, S., Rottmann, K., Sanchez, A., Nash, A., Urbach, L., Kakarala, V., Singh, R., Ranganath, S., Crist, L., Britan, M., Leeuwis, W., Tur, G., Natarajan, P.: Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages (2022)

[2] Mohammadshahi, A., Nikoulina, V., Berard, A., Brun, C., Henderson, J., Besacier, L.: SMaLL-100: Introducing shallow multilingual machine translation model for low-resource languages. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp.

8348–8359. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Dec 2022), https://aclanthology.org/2022.emnlp-main.571

[3] Mohammadshahi, A., Nikoulina, V., Berard, A., Brun, C., Henderson, J., Besacier, L.: What do compressed multilingual machine translation models forget? In: Findings of the Association for Computational Linguistics: EMNLP 2022. pp. 4308–4329. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Dec 2022), https://aclanthology.org/2022.findings-emnlp.317

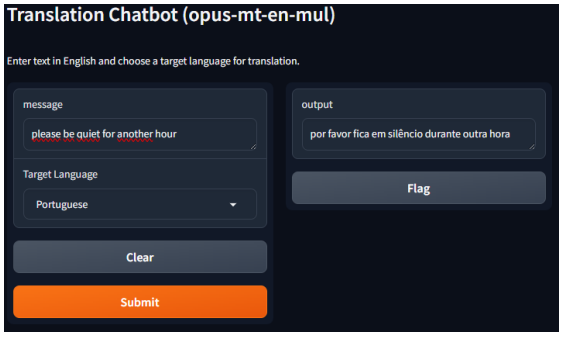
[4] Patrickvonplaten: Helsinki-nlp/opus-mt-en-mul (2020), https://huggingface.co/Helsinki-NLP/opus-mt-en-mul

[5] Tiedemann, J.: The tatoeba translation challenge – realistic data sets for low resource and multilingual MT. In: Proceedings of the Fifth Conference on Machine Translation. pp. 1174–1182. Association for Computational Linguistics, Online (Nov 2020), https://aclanthology.org/2020.wmt-1.139

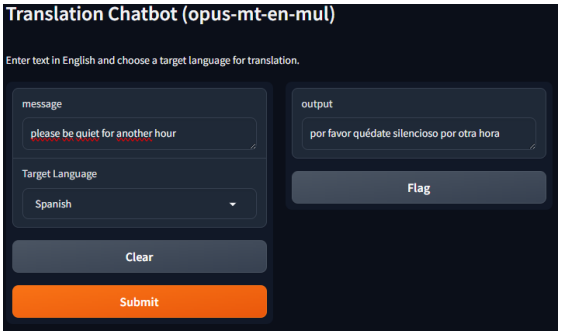
[6] Tiedemann, J., Thottingal, S.: OPUS-MT – building open translation services for the world. In: Proceedings of the 22nd Annual Conference of the European Association for Machine Translation. pp. 479–480. European Association for Machine Translation, Lisboa, Portugal (Nov 2020), https://aclanthology.org/2020.eamt-1.61

A Aplicação

Figura A.1: Modelo “opus-mt-tc-big-en-pt”



(a) Inglês -> Português



(b) Inglês -> Espanhol

Figura A.2: Modelo “opus-mt-en-mul”

(a) Inglês -> Português

(b) Espanhol -> Português

Figura A.3: Modelo “small-100”