Avaliar modelos *few-shot* no *dataset* MASSIVE Português

Luiza Higino e Thiago Martins

Dezembro 2022

Resumo

O *dataset* MASSIVE foi desenvolvido para uma tarefa de compreensão de linguagem natural (NLU, do inglês *Natural Language Understanding*), cujo principal objetivo é transformar texto em requisições para assistentes virtuais. No artigo de introdução do MASSIVE, os autores utilizaram o *dataset* e fizeram *fine-tunning* de modelos de linguagem para avaliação de *slots* e intenções. Neste projeto, entretanto, o *dataset* MASSIVE é aplicado para identificação de cenários em modelos *few-shot* (sem *fine-tunning*). A maior contribuição, portanto, está no desenvolvimento dos *prompts*. Os resultados mostraram que os modelos *few-shot* como Da Vinci 002 e 003 apresentam melhor desempenho nessas tarefas, com acurácia acima de 55% para os três prompts testados.

1 Introdução

As assistentes virtuais têm sido impulsiondas por grandes empresas de tecnologia que já possuem modelos disponíveis em várias línguas e dispositivos, como computador, celular ou até com hardware dedicado. Segundo [3], há uma estimativa de que aproximadamente 120 milhões de pessoas usam assistentes virtuais todos os dias apenas nos Estados Unidos da América. Dessa forma, milhões de dado de requisições às assistentes virtuais são coletados, e estes devem ser transformados corretamente na tarefa que se deseja realizar.

Neste contexto, existem duas tarefas principais de processamento de linguagem natural por assistentes virtuais. A primeira é compreensão da lingugem falada (SLU, do inglês *Spoken Language Understanding*) e a segunda é a compreensão da linguagem natural (NLU, do inglês *Natural Language Understanding*). As tarefas de SLU fazem parte da transformação de áudio em texto, enquanto as tarefas de NLU transformam o texto em intenções (Figura 1). O *dataset* MASSIVE [2], utilizado neste estudo, tem como objetivo auxiliar o treinamento de modelos para NLU. MASSIVE é o acrônimo para *Multilingual Amazon Slu resource package (SLURP) for Slot-filling, Intent classification, and Virtual assistant Evaluation*. Dessa forma, na Figura 1, pode-se identificar *wheather* como

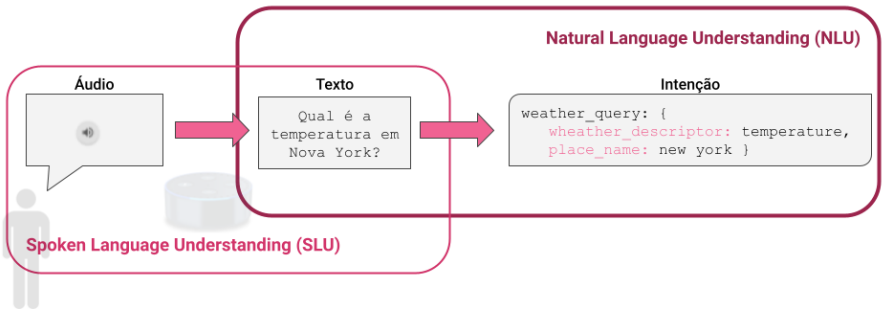

Figura 1: Áreas para interpretação de linguagem para interação com assistentes virtuais

cenário, *wheather query* como intenção, *wheather descriptor* e *place name* como os *slots*.

No artigo de introdução do MASSIVE são apresentados os resultados de acurácia na definição intenções e *slots* em modelos pré-treinados que ainda foram submetidos à *fine-tunning* com o *split* de treino. Todavia, proposta deste projeto final é criar *prompts* com os exemplos do *dataset* MASSIVE e verificar como modelos *few-shot* desempenham a tarefa de definição de cenários.

2 Metodologia

A metodologia para desenvolvimento deste projeto contou com a leitura do artigo de introdução do MASSIVE e familiarização com o *dataset*. Em seguida, *prompts* foram criados usando exemplos de treino do *dataset*, enquanto o *split* de validação foi utilizado para o último exemplo, cujo cenário deveria ser determinado. Os *prompts* foram separados em 3 classes:

- Aleatório com classes diversas (10 exemplos);
- Aleatório (10 exemplos);
- Fixo (15 exemplos).

Os *prompts* continham a definição da tarefa + os possíveis cenários + exemplos de comandos enviados à assistentes virtuais com a definição do cenário ao qual cada exemplo pertencia. Entretanto, foi necessário traduzir os nomes dos 18 cenários para criação dos *prompts*, uma vez que estes eram números no *dataset* original. Dessa forma, modelos *few-shot* foram utilizados para dizer o cenário ao qual o último exemplo pertencia. Os modelos utilizados foram:

- BERTimbau (110 M parâmetros) [6];
- DistilBERT (66 M parâmetros) [4];

- GPT-J (06 B parâmetros) [7];
- FLAN-T5 (780 M parâmetros) [1];
- Da Vinci 002 [5];
- Da Vinci 003 (recém lançado).

Todo projeto foi desenvolvido no ambiente Colab Research (Google Inc.) e utilizou APIs da Hugging Face e da Open AI para acesso aos modelos.

3 *Dataset*

O *dataset* MASSIVE é composto por diversas línguas, entretanto usamos os exemplos em Português de Portugal (pt-PT). Neste caso, são 11514 exemplos de treino, 2033 exemplos de validação e 2974 exemplos de teste. Cada um dos exemplos é composto por um dicionário igual ao apresentado abaixo.

```
   'id': '1',
'locale': 'pt-PT',
'partition': 'train',
'scenario': 16,
'intent': 48,
'utt': 'acorda-me às nove da manhã na sexta-feira',
'annot_utt': 'acorda-me às [time : nove da manhã] na [date : sexta-feira]',
'worker_id': '14',
'slot_method': 'slot': ['time', 'date'],
'method': ['localization', 'translation'],
'judgments': 'worker_id': ['6', '8', '12'],
'intent_score': [1, 1, 1],
'slots_score': [1, 1, 2],
'grammar_score': [4, 4, 4],
'spelling_score':[1, 1, 2],
'language_identification': ['target', 'target', 'target']
```

O interesse do *dataset* MASSIVE é auxiliar no treinamento de modelos para definição de intenções e slots, principalmente. Contudo, há 60 opções de intenções e 55 opções de *slots* no *dataset*, o que aumenta consideravelmente a quantidade de classes com as quais os modelos *few-shot* teriam que escolher a opção correta. Dessa forma, foram utilizados os cenários (que possuem apenas 18 possibilidades) para avaliação dos modelos. A Tabela 1 mostra qual a distribuição dos cenários no *dataset*, e destaca quais são os cenários com maior quantidade de exemplos.

Em [2], os modelos utilizados foram *fine-tunned* com os *splits* de treino e validação do *dataset* para definição das intenções, dos *slots* e de ambos. Com essa metodologia foi possível atingir 62.9% de acurácia média com o mT5 base t*ext-to-text*, 61.2% com mT5 base *encoder only* e 70.6 XLM-R base para classificação de *intenções zero-shot* no *split* en-EN.

Cenário	Treino	Validação	Teste
Social	391	68	106
Transporte	571	110	124
Agenda	**1688**	**280**	**402**
Tocar/jogar	**1377**	**260**	**387**
Notícias	503	82	124
Hora e data	402	73	103
Recomendações	433	69	94
Email	953	157	271
IoT	769	118	220
Geral	652	122	189
Áudio	290	35	62
Listas	539	112	142
Perguntas e respostas	**1183**	**214**	**288**
Culinária	211	43	72
Delivery	257	44	57
Música	332	56	81
Alarme	390	64	96
Tempo	573	126	156
TOTAL	**11514**	**2033**	**2974**

Tabela 1: Exemplos de cenários em cada *split* do MASSIVE pt-PT.

Os cenários existentes no *dataset* original são todas em inglês, mesmo para as sentenças em português. Por isso, a tradução do nome das classes foi feita manualmente. No entanto, alguns cenários em inglês possuem mais de uma tradução em português, como é o caso de "play", e em outros casos como no do cenário "qa" a sigla não existe em português. Esta seria uma sigla para "Question and Answer", mas não existe uma sigla conhecida em português para "Perguntas e Respostas", e por isso a tradução do cenário ficou: "perguntaserespostas", tendo em vista que os modelos preveem apenas uma palavra.

Outra dificuldade encontrada para traduzir o as classes referiu-se aos caracteres especiais da língua portuguesa, como: "ç" e " ". Na maioria dos casos, os melhores modelos conseguiam prever corretamente o nome de classes com este tipo de caracteres. Por outro lado, no modelo FLAN-T5 foi observado uma falha com o caracter especial "ú", contido no nome da classe "música". Esta falha fazia com que o modelo previsse sempre "msica" ao invés de "música". Para contornar este caso, o nome da classe música foi substituído por "musica", sem acento.

4 *Prompts*

Para os experimentos, três tipos diferentes de *prompt* foram criados. Mesmo assim, a única variação entre eles se dá na definição dos exemplos. O cabeçalho

do *prompt* era o mesmo nos 3 casos:

```
    Defina o cenário para o último exemplo abaixo.
    Cenários:  social, transporte, agenda, tocar/jogar, notícias, horaedata, recomen-
dação, email, iot, geral, áudio, listas, perguntaserespostas, culinaria, retirada, mu-
sica, alarme, tempo
```

Para o *prompt* totalmente aleatório, foi considerado que a distribuição dos exemplos seguia a distribuição das classes do conjunto de treino. Desta forma, classes que aparecem mais no conjunto de treino, tendem a aparecer mais no *prompt* aleatório, o que, de certa forma, pode dar um favorecimento no entendimento do modelo e, por consequência, na previsão destas classes.

Já o *prompt* aleatório com classes distintas tem como objetivo aumentar a diversidade de exemplos para que o modelo aprenda mais sobre o maior número de classes possíveis, e assim, obtenha uma maior porcentagem de acerto no maior número possível de classes. Foi então que surgiu outra questão sobre o *prompt*, que seria sobre a quantidade de exemplos a serem utilizados no *prompt*.

Nesta etapa foram realizados testes com diferentes números de exemplos em diferentes modelos. No caso do *Prompt* Fixo, os exemplos foram escolhidos a dedo visando a maior acurácia possível. Para isso, alguns testes foram realizados com as classes que mais confundem os modelos, e durante esse estudo, foi possível verificar que o nome de algumas classes, que foram traduzidas para o português, poderiam dificultar a tarefa dos modelos *few-shot*

4.1 Prompt Aleatório com Classes Diversas

```
Exemplo 1
Comando:  liga e pede uma pizza para entrega
Cenário:  retirada

    Exemplo 2
Comando:  toque música sufi
Cenário:  tocar/jogar

    Exemplo 3
Comando:  porque é que pessoas que passam fome continuam a ter filhos
Cenário:  geral

    Exemplo 4
Comando:  pode remover-me a última lista
Cenário:  listas

    Exemplo 5
Comando:  silenciar
Cenário:  áudio

    Exemplo 6
Comando:  enviar mail de alerta para sandra
Cenário:  email

    Exemplo 7
Comando:  agendar a minha reunião com o sr.  joão armando amanhã à uma da tarde de-
pois do almoço
Cenário:  agenda
```

Exemplo 8
Comando: qual é a população do alabama
Cenário: perguntaserespostas

Exemplo 9
Comando: qual é o nome da música que está a tocar
Cenário: musica

Exemplo 10

Comando: tomo o meu café preto

Cenário: iot

4.2 Prompt Aleatório

Exemplo 1
Comando: coloca folk como favorito
Cenário: tocar/jogar

Exemplo 2
Comando: o alarme programado para as nove horas da noite de quinta-feira
Cenário: alarme

Exemplo 3
Comando: encontre-me as notícias mais populares do dia
Cenário: notícias

Exemplo 4
Comando: que reuniões estão disponíveis em março
Cenário: agenda

Exemplo 5
Comando: diminui volume
Cenário: áudio

Exemplo 6
Comando: em que dia vai estar ensolarado
Cenário: tempo

Exemplo 7
Comando: enumerar todos os eventos do meu calendário de março
Cenário: agenda

Exemplo 8
Comando: eu gosto de qualquer coisa do tiago nacarato
Cenário: musica

Exemplo 9
Comando: há um cozinheiro de chili este fim-de-semana
Cenário: agenda

Exemplo 10

Comando: escrever email para meu colega de trabalho para submeter a tarefa amanhã

Cenário: email

4.3 Prompt fixo

Exemplo 1
Comando: quero ouvir algumas músicas
Cenário: tocar/jogar

```
    Exemplo 2
Comando:  abre o rádio
Cenário:  tocar/jogar

    Exemplo 3
Comando:  toca todas as minhas músicas latinas favoritas
Cenário:  tocar/jogar

    Exemplo 4
Comando:  diz me os detalhes do tempo para dia quinze de março
Cenário:  tempo

    Exemplo 5
Comando:  qual a temperatura para a semana toda
Cenário:  tempo

    Exemplo 6
Comando:  há possibilidade de haver neve quarta-feira
Cenário:  tempo

    Exemplo 7
Comando:  lembrar de enviar e-mail para a minha irmã
Cenário:  agenda

    Exemplo 8
Comando:  há alguma coisa importante para hoje
Cenário:  agenda

    Exemplo 9
Comando:  adicionar evento ao calendário
Cenário:  agenda

    Exemplo 10
Comando:  reenviar um email à susana
Cenário:  email

    Exemplo 11
Comando:  vamos responder a este email
Cenário:  email

    Exemplo 12
Comando:  qual foi o último email que eu recebi
Cenário:  email

    Exemplo 13
Comando:  quero rumores sobre celebridades
Cenário:  notícias


    Exemplo 14

Comando:quais foram as cidades afetadas pelo tremor de terra

Cenário:  notícias
```

5 Resultados

A acurácia de cada um dos modelos quando submetidos aos *prompts* definidos está apresentada na Tabela 2.

Os modelos Da Vinci apresentaram as melhores acurácias pra os três casos, indicando que seria o melhor modelo *few-shot* para desempenho dessa tarefa. Por outro lado, a observação do desempenho do BERTimbau, DistilBERT,

Modelo	Classes diversas	Aleatório	Fixo
BERTimbau	7.60	7.62	9.44
DistilBERT	6.20	6.20	9.98
GPT-J	3.70	8.80	13.23
FLAN-T5	12.40	17.17	21.00
Da Vinci 002	55.6	65.17	59.86
Da Vinci 003	**55.9**	**67.24**	**70.10**

XLM-R base acurácia na avaliação *zero-shot* pós *fine-tunning* no *split* en-EN para **intenções** 70.60%

Tabela 2: Acurácia (%) dos modelos para cada um dos *prompts*.

FLAN-T5 e do GPT-J pode indicar que *prompts* melhores elaborados (com alta supervisão humana) contribuem para uma maior acurácia.

Uma análise mais profunda sobre o desempenho dos modelos foi realizada para o *prompt* fixo. Neste caso, foram avaliados os modelos GPT-J, Da Vinci 002 e Da Vinci 003 (Figuras 2-4). Na legenda de cada uma das barras está a classe correta, à esquerda, e a previsão do modelo, à direita.

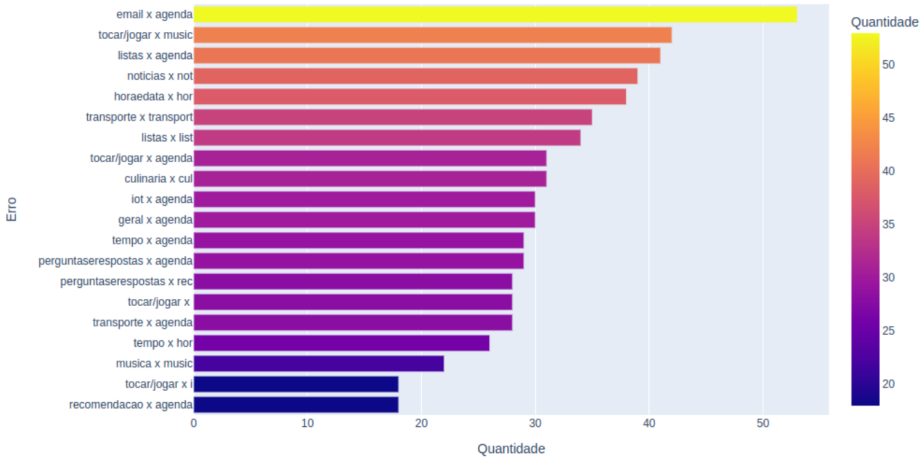

Figura 2: Erros mais comuns GPT-J (Prompt fixo)

Com a Figura 2, percebe-se que as classes que causam mais confusão para este modelo foram email e agenda, o que não é esperado. Pode-se observar também que em 7 dos 20 maiores erros, o modelo previu a classe agenda. E, além disso, tendo em vista que a classe email e a classe tocar/jogar foram utilizadas no *prompt*, podemos dizer que o GPT-J não conseguiu aprender com a quantidade de exemplos utilizada.

As classes mais confundidas pelo Da Vinci 002 foram:

- perguntas e respostas x notícias: provavelmente as perguntas sobre notícias confundiram o modelo;

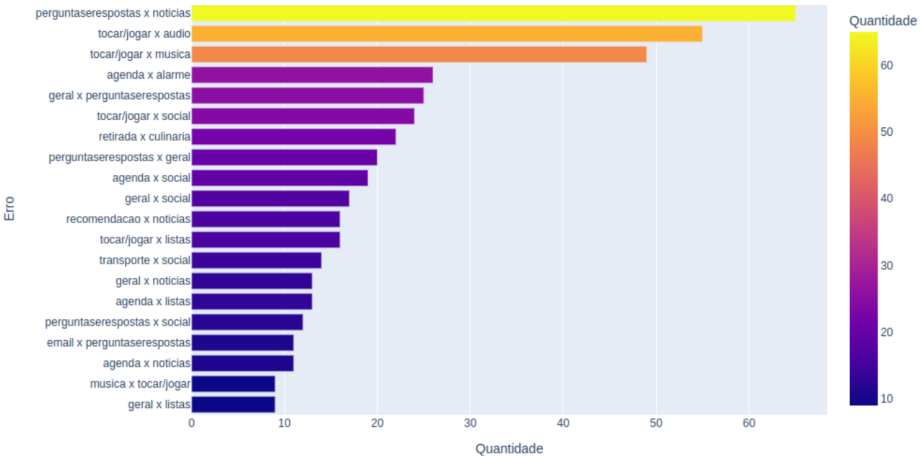

Figura 3: Erros mais comuns Da Vinci 002 (Prompt fixo)

- tocar/jogar x áudio: Essas classes são muito difíceis de diferenciar, pois, os comandos de áudio são solicitações de pausar e mudar música, enquanto os comandos de tocar/jogar são o de tocar uma música e/ou jogar algum jogo. A diferença entre tocar uma música e mudar uma música é muito sutil;
- tocar/jogar x musica: a diferença entre essas classes também é sutil, pois, a classe de música agrupa perguntas relacionadas à música, banda ou cantor.

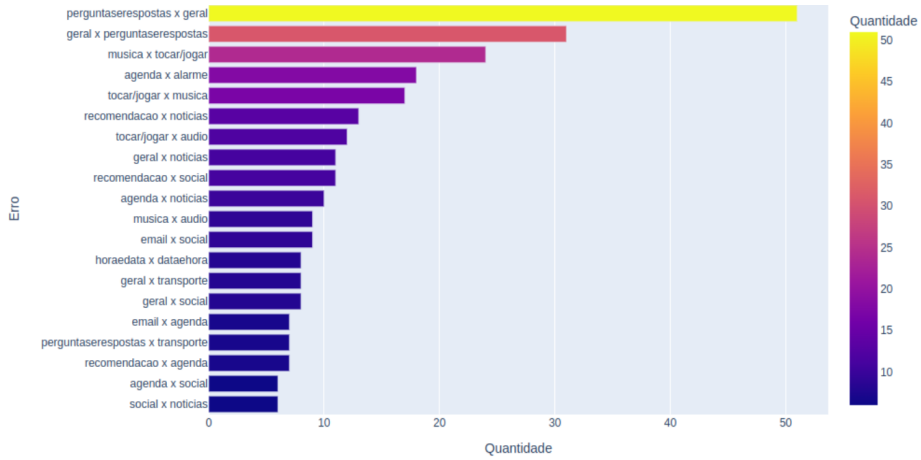

Figura 4: Erros mais comuns Da Vinci 003 (Prompt fixo)

Já no caso do Da Vinci 003 (Figura 4), as duas classes mais confundidas

foram perguntas e respostas x geral. Provavelmente isso aconteceu porque não haviam exemplos dessas duas classes no *prompt*, o que indica que os exemplos no *prompt* ajudaram o modelo a conhecer melhor as classes utilizadas, e que perguntas sobre temas genéricos podem facilmente serem confundidas com o tema geral e vice-versa. Em terceiro lugar ficaram as classes música e tocar/jogar, similar ao já comentado para o Da Vinci 002.

6 Conclusões

Conclui-se que foi possível avaliar os modelos *few-shot* com o *dataset* MASSIVE, usando API da Hugging Face e OpenAI. Além disso, dentre os seis modelos testados, o Da Vinci 003 apresentou o melhor resultado na tarefa de classificação de cenários. Alguns dos modelos propostos no inicio do projeto (XGLM, BLOOM, UL2-20B) não apresentaram resultados significantes ou API não funcionou corretamente para realização dos testes. A tentativa de montagem dos *prompts* foi valiosa para compreender a complexidade de ensinar uma tarefa à uma máquina em poucos passos (o que torna a acurácia do Da Vinci ainda mais impressionante). As traduções e resultados de acurácia abaixo de 25% para os modelos menores indica que a capacidade de aprendizado aumenta com a quantidade de parâmetros dos modelos, entretanto o FLAN-T5 que possui menos parâmetros teve acurácia melhor que o GPT-J. Isso pode estar relacionado aos textos usados no treinamento do FLAN-T5, que englobava cerca de 60 línguas.

7 Trabalhos futuros

Trabalhos futuros podem explorar ainda mais o *dataset* MASSIVE em pt-PT, porém fazendo *fine-tunning* de outro modelos treinados em português, como o GPTimbau, ou até desenvolvimento de *prompts* melhores.

Referências

[1] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[2] Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages, 2022.

[3] Carlos Granell, Paola G Pesántez-Cabrera, Luis M Vilches-Blázquez, Rosario Achig, Miguel R Luaces, Alejandro Cortiñas-Álvarez, Carolina Chayle,

and Villie Morocho. A scoping review on the use, processing and fusion of geographic data in virtual assistants. *Transactions in GIS*, 25(4):1784–1808, 2021.

[4] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[5] Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. Prompting gpt-3 to be reliable. *arXiv preprint arXiv:2210.09150*, 2022.

[6] Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. Bertimbau: pretrained bert models for brazilian portuguese. In *Brazilian conference on intelligent systems*, pages 403–417. Springer, 2020.

[7] Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.