

Labo LAMA-WeST

Intelligence artificielle
Traitement de la langue naturelle
Web sémantique



Génération de texte

INF8460 - Traitement automatique de la langue naturelle

Gaya Mehenni

Polytechnique Montréal

11 novembre 2024

Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



- ▶ Modèles auto-regressifs permettant de générer de nouveaux mots
- ▶ À chaque étape, le modèle prédit, en fonction du contexte précédent et de ses connaissances, le prochain mot
- ▶ Les connaissances du modèle sur la langue et sur le monde sont intégrées dans les poids de son architecture durant une phase de pré-entraînement



- ▶ Compagnie derrière plusieurs librairies à source ouverte facilitant l'utilisation de modèles de langage comme *transformers*, *tokenizers* et *datasets*
- ▶ <https://huggingface.co/>
- ▶ *transformers* : Librairie de traitement de la langue naturelle permettant de charger et d'utiliser des modèles publics



Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



- ▶ 1. Importer les classes nécessaire de la librairie
- ▶ 2. Choisir le modèle (identifiant sur HuggingFace)
- ▶ 2. Charger le modèle et le segmenteur (tokenizer)



```
from transformers import AutoTokenizer, AutoModelForCausalLM

model_name = 'microsoft/Phi-3.5-mini-instruct'

tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)
```



Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



```
text = "The capital of France is"  
inputs = tokenizer(text, return_tensors="pt")
```

```
# `inputs` est un dictionnaire contenant les clés suivantes :  
# input_ids : Indices des jetons  
# attention_mask : Masque binaire indiquant  
# si le jeton doit être pris en compte (utile pour le padding)
```



La plupart des modèles, celui imposé dans le TP4 par exemple, sont entraînés sur un format de conversation. Par exemple avec le format suivant :

```
"""/system/></system/>
You are a helpful assistant.</end/>
</user/>
Question?</end/>
</assistant/>"""
```

Il est donc important de respecter ce format dès qu'on utilise ou entraîne ce modèle.



Le meilleur moyen d'appliquer le bon format de conversation est d'utiliser la méthode `apply_chat_template` du tokenizer.

```
input = 'What is the capital of France ?'  
# tokenize est mis à False pour qu'on voit  
# le format de conversation. Normalement,  
# on le laisserait à True pour appliquer  
# le format de conversation et segmenter  
# automatiquement  
  
input_template = tokenizer.apply_chat_template([  
    {'role': 'user', 'content': input}  
], tokenize=False)  
  
print(input_template)  
# </user/>  
# What is the capital of France ?</end/>  
# </endoftext/>
```



Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



La méthode `generate()` permet de générer de nouveaux jetons en fonction d'un texte en entrée :

```
# Segmentation  
# `The capital of France is`  
inputs = tokenizer(text, return_tensors="pt")  
  
# Génération  
outputs = model.generate(**inputs, max_length=15)  
print(outputs[0]) # `Paris`
```



La méthode 'generate()' permet d'appliquer plusieurs algorithmes de génération comme *Beam Search*, *Top-k sample*,

Beam search

```
outputs = model.generate(inputs["input_ids"], max_length=50,  
                          num_beams=5, no_repeat_ngram_size=2)
```

Sampling

```
outputs_sampling = model.generate(inputs["input_ids"], max_length=50,  
                                  do_sample=True, top_p=0.95,  
                                  temperature=0.7)
```

Top-K

```
outputs_top_k = model.generate(inputs["input_ids"], max_length=50,  
                               do_sample=True, top_k=50)
```

Plus d'information ici



Introduction

Chargement d'un modèle

Segmentation des données

Génération de texte

Prompting



Permet d'indiquer au modèle, à partir d'exemples précédents, comment générer les prochains jetons

Classifie le sentiment des phrases suivantes :

Phrase : J'adore ce film !

Sentiment : Positif

Phrase : Cette journée est horrible.

Sentiment : Négatif

Phrase : Le temps est nuageux aujourd'hui.

Sentiment : Neutre

Phrase : Ce restaurant est vraiment décevant.

Sentiment :



Permet d'expliquer au modèle le raisonnement derrière une réponse pour le guider dans sa génération

Résous le problème suivant étape par étape :

Problème : Si 5 pommes coûtent 2 euros, combien coûtent 12 pommes ?

Étape 1 : Déterminons le coût d'une pomme

5 pommes = 2 euros

1 pomme = $2 / 5 = 0,40$ euro

Étape 2 : Calculons le coût pour 12 pommes

*# 12 pommes = $12 * 0,40 = 4,80$ euros*

Réponse : 12 pommes coûtent 4,80 euros.

Résous le problème suivant étape par étape :

Problème : Si 8 pommes coûtent 2 euros, combien coûtent 24 pommes ?

