**NLP Basics & End Tasks**

Confluence : all the doc I wrote

ChatGPT usecases :

* Generate Jira tickets following a title and a template
* Help me formalize emails
* Check the installed extensions

Table des matières

[1. Basics in NLP : 2](#_Toc146544226)

[1. Theory 2](#_Toc146544227)

[2. Libraries : 2](#_Toc146544228)

[3. Finetuning tutorials 2](#_Toc146544229)

[2. Embeddings: 3](#_Toc146544230)

[1. Static word embeddings vs contextual 3](#_Toc146544231)

[2. Word embeddings : 3](#_Toc146544232)

[3. Large language models 3](#_Toc146544233)

[1. What are autoregressive models ? 3](#_Toc146544234)

[2. Autoregressive vs AE vs seq2seq : 4](#_Toc146544235)

[3. What is a Language model? 4](#_Toc146544236)

[4. Finetuning vs feature based 4](#_Toc146544237)

[5. Teacher forcing : 5](#_Toc146544238)

[6. why there is no preprocessing step for training BERT? 5](#_Toc146544239)

[7. Parameters of LLMs (temperature, top-k, top-p…): 5](#_Toc146544240)

[8. Language generation strategies 5](#_Toc146544241)

[9. Floating point precision formats and their representation/pytorch : 6](#_Toc146544242)

[10. LORA (Low-Rank Adaptation): 6](#_Toc146544243)

[11. RLHF 6](#_Toc146544244)

[2. End tasks 6](#_Toc146544245)

[1. Information extraction & NER (full doc on confluence) 6](#_Toc146544246)

# Basics in NLP :

## Theory

* Rule based / statistical NLP [Natural Language Processing (NLP) with Python — Tutorial – Towards AI](https://towardsai.net/p/nlp/natural-language-processing-nlp-with-python-tutorial-for-beginners-1f54e610a1a0)
* RNN & LSTMs : [Aller plus loin en deep learning avec les réseaux de neurones récurrents (RNNs) - Devoteam France](https://france.devoteam.com/paroles-dexperts/aller-plus-loin-en-deep-learning-avec-les-reseaux-de-neurones-recurrents-rnns/#:~:text=Le%20LSTM%20(Long%20Short%2DTerm,la%20m%C3%A9moire%20courte%20des%20RNNs.)
* Check LLMs section for LLMs theory

## Libraries :

* List of interesting NLP repositories
* Flair
* [flair/TUTORIAL\_TAGGING\_OVERVIEW.md at master · flairNLP/flair (github.com)](https://github.com/flairNLP/flair/blob/master/resources/docs/TUTORIAL_TAGGING_OVERVIEW.md)
* [Flair Basics (cavar.me)](http://damir.cavar.me/pynotebooks/Flair_Basics.html)
* Spacy 101 : [spaCy 101: Everything you need to know · spaCy Usage Documentation](https://spacy.io/usage/spacy-101)
* HF Transformers : [🤗 Transformers (huggingface.co)](https://huggingface.co/docs/transformers/index)
* LLMs:
  + Text generation webui
  + Llama-cpp-python
  + Exllama
  + Langchain
* Transformers library
  + Paper : [This Aged Like Fine Milk - YouTube](https://www.youtube.com/watch?v=1m8ORMMeI38)
  + Tutorials: [🤗 Transformers (huggingface.co)](https://huggingface.co/docs/transformers/index)
  + Every model in the library is fully defined by three building blocks shown in the diagram in Figure 2: (a) a tokenizer, which converts raw text to sparse index encodings, (b) a transformer, which transforms sparse indices to contextual embeddings, and (c) a head, which uses contextual embeddings to make a task-specific prediction. Most user needs can be addressed with these three components.
* AllenNLP
* Flair
* Stanza.

## Finetuning tutorials

(finetuning to check later if it is useful) :

* [Natural Language Processing with spaCy & Python - Course for Beginners - YouTube](https://www.youtube.com/watch?v=dIUTsFT2MeQ)
* [How to Fine-Tune BERT Transformer with spaCy 3 | by Walid Amamou | Towards Data Science](https://towardsdatascience.com/how-to-fine-tune-bert-transformer-with-spacy-3-6a90bfe57647)
* [Fine-tune GPT-J: a cost-effective GPT-4 alternative for many NLP tasks (graphcore.ai)](https://www.graphcore.ai/posts/fine-tuned-gpt-j-a-cost-effective-alternative-to-gpt-4-for-nlp-tasks)
* Class imbalanced : [2023.eacl-main.38.pdf (aclanthology.org)](https://aclanthology.org/2023.eacl-main.38.pdf)
* [Resume (CV) Parsing using Spacy 3 | NER Training in Spacy v3 - YouTube](https://www.youtube.com/watch?v=WpaioLNsoGI)
* [NLP+CSS 201 Tutorials | Tutorials for advanced natural language processing methods designed for computational social science research. (nlp-css-201-tutorials.github.io)](https://nlp-css-201-tutorials.github.io/nlp-css-201-tutorials/)
* [Named Entity Recognition in Python for Digital Humanities - YouTube](https://www.youtube.com/playlist?list=PL2VXyKi-KpYs1bSnT8bfMFyGS-wMcjesM)
* [Advanced NLP with spaCy · A free online course](https://course.spacy.io/en)
* [INTRODUCTION TO NAMED ENTITY RECOGNITION — Introduction to Named Entity Recognition (pythonhumanities.com)](https://ner.pythonhumanities.com/intro.html)

# Embeddings:

## Static word embeddings vs contextual

Both embedding techniques, traditional word embedding (e.g. word2vec, Glove) and contextual embedding (e.g. ELMo, BERT), aim to learn a continuous (vector) representation for each word in the documents. Continuous representations can be used in downstream machine learning tasks.

Traditional word embedding techniques learn a global word embedding. They first build a global vocabulary using unique words in the documents by ignoring the meaning of words in different context. Then, similar representations are learnt for the words appeared more frequently close each other in the documents. The problem is that in such word representations the words' contextual meaning (the meaning derived from the words' surroundings), is ignored. For example, only one representation is learnt for "left" in sentence "I left my phone on the left side of the table." However, "left" has two different meanings in the sentence, and needs to have two different representations in the embedding space.

On the other hand, contextual embedding methods are used to learn sequence-level semantics by considering the sequence of all words in the documents. Thus, such techniques learn different representations for polysemous words, e.g. "left" in example above, based on their context.

Source <https://stackoverflow.com/a/62355854>

## Word embeddings :

Source : [From Static Embedding to Contextualized Embedding | by Ted Mei | Medium](https://ted-mei.medium.com/from-static-embedding-to-contextualized-embedding-fe604886b2bc)

# Large language models

## What are autoregressive models ?

An autoregressive language model is a type of Machine Learning model that uses autoregressive techniques to predict the next word in a sequence of words based on the words that have come before it.

the mechanism of language model is to predict the next word given previous words.

Both GPT-2 and ELMO are auto-regressive models (model based on language model). In ELMO corpus is trained on both left-to-right and right-to-left directions. However in GPT-2, corpus is trained only from left to right. GPT-2 has better performance than ELMO due to its Transformer architecture and much larger size.

## Autoregressive vs AE vs seq2seq :

* AR : An AR model learns from a series of timed steps and takes measurements from previous actions as inputs for a regression model, in order to predict the value of the next time step.

AR models are typically used for generation tasks, such as tasks in the domain of natural language generation (NLG), for e.g., summarization. (GPT, GPT-2, GPT-3)

* AE based pretraining does not perform explicit density estimation but instead aims to reconstruct the original data from corrupted input. (BERT)
* Encoder-Decoder/Seq2seq Models : uses both an encoder and decoder. It treats each task as sequence to sequence conversion/generation (for e.g., text to text, or even multimodal tasks such as text to image or image to text). For instance, for text classification, the encoder takes text as input, and the decoder generates text labels instead of classifying them. (XLNET)

Source : [Aman's AI Journal • Primers • Autoregressive vs. Autoencoder Models](https://aman.ai/primers/ai/autoregressive-vs-autoencoder-models/)

Source : [machine-learning-articles/differences-between-autoregressive-autoencoding-and-sequence-to-sequence-models-in-machine-learning.md at main · christianversloot/machine-learning-articles (github.com)](https://github.com/christianversloot/machine-learning-articles/blob/main/differences-between-autoregressive-autoencoding-and-sequence-to-sequence-models-in-machine-learning.md)

The normal Transformer decoder is autoregressive at inference time and non-autoregressive at training time. (Source : <https://datascience.stackexchange.com/a/104188>)

## What is a Language model?

A language model uses machine learning to conduct a probability distribution over words used to predict the most likely next word in a sentence based on the previous entry. There two types : statistical LM and neural LM.

Source : [What are Language Models in NLP? (daffodilsw.com)](https://insights.daffodilsw.com/blog/what-are-language-models-in-nlp)

Series of articles :

ML articles : [christianversloot/machine-learning-articles: 🧠💬 Articles I wrote about machine learning, archived from MachineCurve.com. (github.com)](https://github.com/christianversloot/machine-learning-articles)

## Transformers attention mechanism & complexity

**Flash attention:**

<https://medium.com/@datadrifters/more-more-quadratic-complexity-for-transformers-discover-the-power-of-flash-attention-a91cdc0026ed>

<https://medium.com/@sthanikamsanthosh1994/introduction-to-flash-attention-a-breakthrough-in-efficient-attention-mechanism-3eb47e8962c3>

## Finetuning vs feature based

* Fine-tuning approaches typically only add a single linear layer to a transformer and fine-tune the entire architecture on the NER task.
* Feature-Based Feature-based approaches instead use the transformer only to generate embeddings for each word in a sentence and use these as input into a standard sequence labeling architecture, most commonly a LSTM-CRF (Huang et al., 2015). T

Document-level features. : While NER is traditionally modeled at the sentence-level, transformer based models offer a natural option for capturing document-level features by passing a sentence with its surrounding context.

[2011.06993v2.pdf (arxiv.org)](https://arxiv.org/pdf/2011.06993v2.pdf)

## Teacher forcing :

Using Output as Input in Sequence Prediction

## [why there is no preprocessing step for training BERT?](https://datascience.stackexchange.com/questions/113359/why-there-is-no-preprocessing-step-for-training-bert)

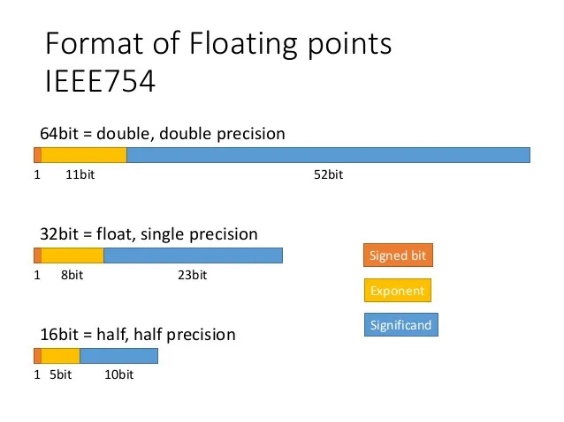
It's not mandatory. Removing stopwords can sometimes help and sometimes not. You should try both.

A case for not using stopwords: Using stopwords will provide context to the user's intent. So when you use a contextual model like BERT, all stopwords are kept to provide enough context information like the negation words (not, nor, never) which are considered to be stopwords.

With BERT you don't process the texts; otherwise, you lose the context (stemming, lemmatization) or change the texts outright (stop words removal).

Source : <https://stackoverflow.com/a/63635644>

## Floating point precision formats and their representation/pytorch :



* <https://medium.datadriveninvestor.com/mixed-precision-training-for-deep-neural-networks-3751f2c88883>
* [Écriture d’un entier positif en base 2 (pixees.fr)](https://pixees.fr/informatiquelycee/n_site/isn_base_2.html)
* [Nombres binaires : Comprendre l'encodage du signe et de la virgule – Buzut](https://buzut.net/cours/computer-science/signes-virgule-precision)
* [The mechanics behind exponent bias in floating point - JavaScript inDepth](https://indepth.dev/posts/1018/the-mechanics-behind-exponent-bias-in-floating-point)
* [What is the Base-10 Number System? (thoughtco.com)](https://www.thoughtco.com/definition-of-base-10-2312365)
* [A Gentle Introduction to 8-bit Matrix Multiplication for transformers at scale using transformers, accelerate and bitsandbytes (huggingface.co)](https://huggingface.co/blog/hf-bitsandbytes-integration)
* [Making LLMs even more accessible with bitsandbytes, 4-bit quantization and QLoRA (huggingface.co)](https://huggingface.co/blog/4bit-transformers-bitsandbytes)

## LORA (Low-Rank Adaptation):

* [Fine-tuning LLMs Made Easy with LoRA and Generative AI-Stable Diffusion LoRA | by xiao sean | Mar, 2023 | Medium](https://xiaosean5408.medium.com/fine-tuning-llms-made-easy-with-lora-and-generative-ai-stable-diffusion-lora-39ff27480fda)

## RLHF

* [Reinforcement Learning from Human Feedback(RLHF)-ChatGPT | by Sthanikam Santhosh | Medium](https://medium.com/@sthanikamsanthosh1994/reinforcement-learning-from-human-feedback-rlhf-532e014fb4ae)
* [2203.02155.pdf (arxiv.org)](https://arxiv.org/pdf/2203.02155.pdf)

# End tasks

## Information extraction & NER

**full doc on confluence**

* IE vs IR vs NER : [Information Extraction and Named Entity Recognition (unice.fr)](https://www.i3s.unice.fr/~tettaman/Classes/WebScience/2014/Cours_IE_WebScience.pdf)
* papers and details : Confluence

### NER :

Libraries:

* Flair
* SpanMarker : [tomaarsen/SpanMarkerNER: SpanMarker for Named Entity Recognition (github.com)](https://github.com/tomaarsen/SpanMarkerNER)
  + All SpanMarker models on the Hugging Face Hub can also be easily used in spaCy
  + They are the best pretrained models (Sep 2023)