4 Natural Language Processing

Note takers: Blanke, Daoud, Duchemin, Gourru, Jhuboo, Jourdan, Lauga, Mercklé, Sandberg, Terreau

Instructor: Franǫis Yvon (LIMSI/CNRS)

[Video on the IBM project debater] Today, machines are capable of amazing, human-level performances.

4.1 Language: a hard nut to crack

Natural Language Processing (NLP) was studied for three main reasons:

- language processing as computation,
- computational psycholinguistics,
- automated processing tools and applications.

A classical approach to automatic speech recognition consisted in the following **pipeline model**.

- Lexical decoding: from a continuous-time audio signal to a discrete sequence of phonetic symbols.
- Orthographic decoding: create words and a sentence out of the phonetic symbols.
- Text normalisation: identify categories of words in the sentence.
- Structure identification: find the dependencies between the words.

The way back is even harder.

An example of tricky sentence "Le cousin de Paul se piquait de bien connaître la ville". It is tricky for several reasons. "Connaître" doesn't have a subject. "Cousin" can mean "cousin" but also a tipula. "Se piquer" could be confused with an insect biting. This is called word sense disambiguation.

More than processing one sequence, it is even harder to handle a span of text of several sentences with coreferences between them. Typically, different mentions of one single entity (*e.g.* one person) make an automatic processing very challenging.

Nowadays, there are a number of NLP tasks that go beyond isolated sentences. Classifying sentences, for instance by tense, mood, polarity etc. Typically, classifying tweets. Finding the structure of a text.

The pipeline model mentioned above often struggles because of arbitrarily long, multi-layered dependencies across the pipeline.

Another typical case of reason of failure is the ambiguity of some words. In politics for example, the chair (organizer) of a conference would sometimes be understood as a chair as the piece of furniture. In French, just think of the word "et" which can easily be confused with "et" or even "hait".

Pipeline model does not work. "It's like building a compiler, but you have only part of the syntax."

- Errors accumulate down the pipe.
- Early decisions require deep analysis.
- Ambiguity is a feature, not a defect (puns!).
- Segmentation ambiguities
  - *gardes* plural or second person
- Lexical ambiguities
- Syntactic ambiguities
  - "N. H. Defends Laconia Law Barring Female Nudity In Supreme Court Ruling"
- Semantic Ambiguities
- Pragmatic Ambiguities
  - Understand that "I'm cold" means "Close the window"

Language is always evolving.

- Phonetic changes and reconfigurations
- New spellings and grammatical constructs
- Lexicalization of new derivatives and compounds

- New senses appearing
-

4.2 The great paradigm shift : towards statistical NLP

With more resources and a more fine grained description of languages, we could get to this nice pipeline scenario. Around 92-93, statistics have progressively been incorporated into NLP. This transition has officially started with a publication in computation linguistic in the context of special issues on corpora based approaches.

Switch from grammar to corpora patterns : the hypothesis is to find ways to process large facets of languages. 2011, Norvig

The first ingredient to moving to statistical language models. Arguments for graduality in language: grammatical rules and judgement can be gradual : for instance house is a noun but home is a "better" noun than house, that has a larger combinatorial power. Similarly, "grièvement" only applies to specific contexts, while "gravement" can have much more applications and would be preferred as both mean the same. Finally, Human brain appears to be sensible to frequency : we recognize frequent word quicker. These are arguments to go toward statistical treatment of natural language.

A second ingredient is the collection of a large corpus of relevant data. Linguistic Data Consortium - LDC () : catalog of corpora, resources for annotation, rare languages, see LREC.

The third ingredient is the development of NLP challenges by funding agencies. They focused on having strong methodological construction of tasks.

- describe the task exactly
- what is given to participants (computational resources)
- what is the metric
- distribute test data for final evaluation

(Repository to track the progress in Natural Language Processing (NLP)). People tend to participate to these challenges for : access to data, and access funding. These have been highly influential to move to statistical methods that were, most of the time, the most accurate approaches.

4.3 Discovery of statistical method : the effectiveness of simple models

The simplicity of those models comes from the fact that they do not need to know any of the rules of a language and simply works from statistic measure (for example how likely some

words will be written close together). In the following, we describe three important applications in NLP of this statistical viewpoint.

Speech recognition : a recurrent problem is how to decide the correct sentence to write for a given recorded sample, e.g "danser, dansés, dansé, dansée". How to compare sentences ?

Language models, simple yet effective (n-grams), with L the length of a sentence, V the vocabulary space.

$$P(w_1, ., w_L) = \prod_i^L p(w_i|w_1...w_{i-1}) = \prod_i^L p(w_i|w_{i-n+1}...w_{i-1}) \tag{4.1}$$

This technique allows to process omre than words such as letters, speech

Information Retrieval: Bag-of-words
→ Main idea: *"Turn a document into a vector."*
Each document is embedded as a vector $d \in \mathbb{R}^{|\mathcal{V}|}$ with $d^\top = (x_1, \ldots, x_{|\mathcal{V}})$. A typical choice for x_i is

- $x_i = \frac{N(w_i \in d)}{l_d}$ where $N(w_i \in d)$ is the number of times the word w_i appears in the document d and l_d is the number of words in the document d.
- $x_i = TF - IDF(w_i)$ (Term Frequency(TF) — Inverse Dense Frequency(IDF)).

This embedding method allows to compare two documents d and d' using several measures as scalar product, cosine sim, standard distances.

Computational lexicography
→ Main idea: *"You shall know a word by the company it keeps."*
We compute semantic relationship from distributional observations: shared contexts imply semantic relatedness.

Considering a fixed vocabulary, $\mathcal{V} = \{v_1, \ldots, v_{|\mathcal{V}|}\}$. For any word w, $r(w) \in \mathbb{R}^{|\mathcal{V}|}$ is the vector where the i-th entry counts the number of time the word v_i is a neighbor of the word w in the corpus. Then, the distance between two words w and w' is given by

$$\text{dist}(w, w') \propto r(w)^\top r(w').$$

4.4 From empirical methods to machine learning techniques

Supervised classification Resolving ambiguities by building trees. This can be turned to a simpler problem : finding dependencies, which are binary decision that can be solved using ML methods. Difficult step : find the good features to describe the data and the problem (context

Bag of Words, position in the sentence, sentence type). Other examples that can also be solved using classification tools are:

- word sense disambiguation,
- sentence segmentation
- co-reference resolution
- sentiment analysis
- ...

Results in from 1993-2010 can be summed up as: find a problem (word sense ambiguity,..), formulate it as a classification problem and use ML tools.

More ML related topics also emerged, such as high dimension, metrics, high number of classes that can even be organized in hierarchies.

ML was also successful in more applications cases than simple classification, e.g. parsing trees. These can be learnt step by step. Most exactly, what is learned is the sequence of actions needed (analogous to robotic movements). Action are not observed, some different sets of actions can lead to the same parsing tree. Dependency parsing build acyclic set of arcs between words.

No crossing arcs = projectivity(= easier to solve. Non-projectivity is rare in French and english. This allows some fast algorithms: greedy left-right decoding.

Transition-based projective dependency parsing: guaranteed to have an acyclic graph.

Remark: Punctuation is treated as words. But there are markers for the start and the end of a sentence, so that we know when words are usually used at the end (like punctuation). The main task of modeling structure, syntax of the language is to define a way do decide if a word or a sentence is better than an other. If one notes A and B two sentences, one can introduce the equivalence between A being better than B and a probability $P(A)$ being higher than a probability $P(B)$. This probability $P(sentence)$ is a language model. Such probability can easily be derived from a simple Markov assumption to predict the likelihood of one word based on the preceding words. This Markov assumption is very naïve but is extremely efficient computationally. One call this type of models n-grams and for a sequence of words $(w_1, ..., w_L)$ we define :

$$P(w_1, ..., w_L) = \prod_{i=1}^{L} P(w_i | w_1, \ldots, w_{i-1})$$

New architectures(transformers) are trained to learn which words matter in the history. In the past the importance was fixed. Feed-forward: fixed number of words in history. RNN: older words are gradually forgotten.

4.5 Transformers and self-attention

Compute with heads Compute linear weights:

$$\tilde{D} = \text{softmax}(D/\sqrt{d}) \in [0,1]^T \times [0,1]^T. \tag{4.2}$$

4.6 Evaluating language models

4.6.1 Large Language Models are *very* powerful

Originally used as scoring model for disambiguation tasks.

Will now cover use as text generators. Can use language models for any natural language tasks (Radford et al 2019).

Tasks: Give your model a prompt, and generate next word. Probability over possible next words. Can apply this to lots of different domains, such as translation, or even arithmetic. In the latter case they even are often correct.

The language models are evaluated with a measure called perplexity.

$$\text{PPL}(M) = 2^{\frac{1}{T}} \log_2 P(w[1:T]|M) \tag{4.3}$$

Cross entropy between source and model.

Before NNs language models had fairly bad perplexity (120 nats), now we reach around 6 times lower for models trained on English texts.

Evaluation with linguistic probes. How to evaluate if we learn long-range structural dependencies. Ex: Subject verb agreement. Subject must agree in number with the object, but they can bee far apart in the text. "The keys to the cabinet (are|is) on the table."

Linzen et al, (2016), traine an LM-RNN to predict the verb number. Performance good (1% error rate). Drops slowly with subject-verb distance. Drops slowly with intervening distractors (eg singular words between subject and verb). If instead train a NN to predict next word we get a 10-fold loss in performance. In complex cases, more direct form of training signal is needed to learn the correct structure.

4.6.2 Algorithms for text generation

Greedy Search At each step, the most likely word given the past is chosen.

Ancestral sampling

$$w_0 =< s > \tag{4.4}$$

$$w_t \sim P(w|w(t' \leq t)) \tag{4.5}$$

Nucleus Sampling

Language model (de)generation In practice, these text generation algorithms end up generating loops, even though they are syntaxically consistent.

High probability sentences do not resemble human productions.

- Too many repetitions.
- High-frequency tokens are over-represented, and low-frequency ones underrepresented.
- Lack of diversity.
- Lack of global consistency.
- Poorly calibrated posterior distribution.

Action takes place in very high-dimensional space. Predict next step from current vector in this space. Easy to go to "nonsensical" parts of the space. Easy to get caught in loops.

Beam search [with histogram pruning] More improved search algorithms (Wiher et al, 2022).

Better learning losses. - Use label smoothing.

4.6.3 Evaluating LMs with distributional properties

Evaluating Zero-shot/ few-shot behavior.

- Zero-shot learning, No demonstrations. "translate English to French: cheese -> ?"
- One-shot learning, one demonstration,...
-

Current challenges for language modeling.

- Text generation is still difficult.
- Improve efficiency and scalability.
- How to update models as language changes.
- How to avoid models learning hate-speech, and how to remove e.g. private information without having to retrain model, etc. (Stochastic Parrots)

4.7 Transfer learning

4.7.1 Multi-task learning and pretraining

Learning representations for NLP in an unsupervised way (Collobert 2011). Instead of having one benchmark for each task, they say we want to have one system for all tasks, and to learn it in an unsupervised manner.

In 2018 people started to implement this program at scale. Recipe:

1. take huge corpus to train embeddings (unsupervised).
2. Use this representation as features for supervised training on domain specific task.

Popular models include ELMO (Peters et al, 2018) and BERT (Devlin et al 2019).

Elmo is a network made by several stack of bidirectional RNNs. Pass sentence through RNNs back and forth, then passed to next layer. All layers are combined to yield the final representation.

Bert is a transformer, but it is non-causal. It can see the full sentence, and is trained by masking some of the words. In the last layer the goal is to predict the masked words.

BERT, and similar pre-trained encoders, typically give significantly improved performances. A lot of encoders these days are in the form of encoder-decoder pairs, using next word prediction, deshuffling, denoising, or similar techniques to avoid the need for labelling.

Benefits of LM pretraining: - Leverage large corpuses of text in an almost unsupervised way. - Allows for knowledge transfer between domains.

4.8 Multilingual NLP

4.8.1 Introduction

Diversity of languages around the world (see https://www.ethnologue.com/guides). These are divided in language families, which are not equality distributed around the world. The top 25 languages only covers half of the world population. Countries with only one language are the exception, bilingualism (or more) is the norm. Many languages are endangered due to their lack of use in the population from some economics point of view or else. New languages have also been created. This diversity of languages is particularly surprising given that languages have the same origin and humans have the same brain structure. Nevertheless there is a wide variety of linguistic systems.

NLPers should care for several reasons (https://ruder.io/nlp-beyond-english/) such as political/societal, economic, linguistic, Machine Learning, cultural/historical, cognitive. Motivation for using NLP in multilingual setting : usual publications don't even give the language that are studied, as English turned to be the standard ML language.

The typical methods of multilingual NLP are machine translation, multilingual models (mGPT, the multilingual version of GPT, mBERT) and cross-lingual representations and transfer. The

available resources are parallel and comparable corpora (https://opus.nlpl.eu, wikipedia), bilingual dictionaries (https://panlex.org), comparative/typological language documentation (https://wals.info).

Main challenges Multilingual NLP suffers from a large resource unbalance (Joshi et al., 2020, https://arxiv.org/abs/2004.09095). Languages can be clustered into classes having different scales of available (labeled or not) data. While those resources are high for seven languages (0.27%, spoken by 2.5 billions), 2191 languages (88.38%, spoken by 1.2 billions) have no existing resources, such as annotated data for supervised settings. Therefore those languages can't benefit from recent technologies using NLP, e.g. voice command (phone, car, ...).

Nowadays some informal language sentences can mix two languages, e.g. bilingual speaker. New interesting problems arise such as language contact and code-switching. This lead to new tasks: language identification, language transcription and analysis, language translation, CS generation. See for example (Sitaram et al., 2019, https://arxiv.org/abs/1904.00784).

For moderation problem, hateful speeches are sometimes not recovered for low resources languages.

4.8.2 ML models for Machine Translation (ML)

An attempt to handle multilingual data consists in using machine translation.

The first approach used vanilla RNN (Recurrent Neural Network). Encoder decoder systems (seq2seq), go through a first phase of sentence encoding, then recursively generate the translated sentence (the target sentence). The main issue is that all the information of the source sentence need to be encoded in a (memory less) hidden vector. As this is not enough to store all the necessary information, this nice and simple approach fails.

To circumvent this problem, attention mechanisms were proposed (Bahdanau https://arxiv.org/abs/1409.0473, Luong https://arxiv.org/abs/1508.04025). The hidden representation is now a linear combination of the latent representations of the source sentence words. The network is modifying the representation of the whole sentence representation for the current word generation in the target sentence. Additionally, the attention matrix provides, for each generated word in the target language, the relative importance of each word in the source sentence.

This further led to the Transformer, that was initially proposed as a seq2seq model for language translation. The encoder is used for language modeling tasks. In a seq2seq setting, the decoder is using cross attention, i.e. computing attention between target and source sentences, which is not done in the Transformer encoder-only architecture (such as BERT). One main advantage of the multi head attention is the possibility to compute in parallel. See also Popel et al. (2020, https://www.nature.com/articles/s41467-020-18073-9.pdf).

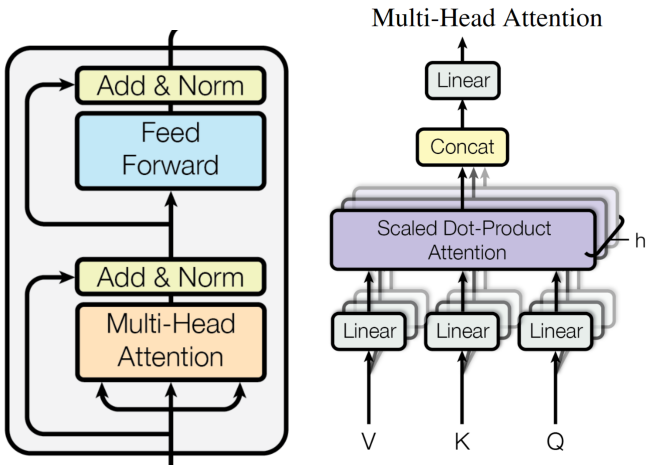

Figure 4.1: Bloc Transformer and multihead attention https://arxiv.org/abs/1706.03762

Multilingual models The usual approach took pairs of language. Now multilingual models translate from various languages to various languages with a single model (Firat et al., 2016, https://aclanthology.org/N16-1101.pdf. Johnson et al., 2017 https://arxiv.org/abs/1611.04558). This needs to build some joint representations of word pieces. A lot of studies are conducted on multilingual representations, i.e. words in several languages represented in the same low dimension continuous space. This assume that there is a stable notion of word across languages. This can be done using a Transformer (Conneau and Lample 2018, https://arxiv.org/abs/1901.07291)

Some crucial properties of Neural MT: segmentation in sentences and words, spelling and grammatical correction/normalization, grammatical parsing, sentence simplification.

Even bad MT is more useful than you think. MT translates artificial training data (text+labels) into other languages.

Universality of languages: X-lingual transfer learning (Yarowsky et al., 2001, https://aclanthology.org/N01-1026.pdf). The four main steps are:

- Automatic word alignment of parallel sentences
- PoS tag source data
- Project tags via alignment links
- Use of a PoS tagger with projected data

To obtain multilingual representations one should compute embeddings such that mutual translations nearest neighbours

- bilingual skip-gram
- X-lingual word space alignment with bilingual dictionary

- multilingual sentence representation via multilingual translation
- joint encoding/decoding with round-trip-translation

XLM (Lample and Conneau, 2019, https://arxiv.org/abs/1901.07291) learns multilingual contextual embeddings.

Conclusion Toward deep language understanding ? The language models are currently scaling to enormous datasets, thanks to more resources in term of materials, money and working force, that are dedicated to the field. A lot of what is done is based on many heuristics : it requires to go toward better optimizations for these huge Language models. Additionally, these languages do not incorporate knowledge. Finally, evaluation system are not properly built, and might prevent from getting the limitations of existing approaches. This is particularly difficult with text : how to evaluate if a sentence is “good” ?

Bibliography

Shipra Agrawal and Nikhil R Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 989–1006, 2014.

Shipra Agrawal and Randy Jia. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. *Advances in Neural Information Processing Systems*, 30, 2017.

Shipra Agrawal, Vashist Avadhanula, Vineet Goyal, and Assaf Zeevi. Mnl-bandit: A dynamic learning approach to assortment selection. *Operations Research*, 67(5):1453–1485, 2019.

Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21, 2008.

Francis R. Bach and Eric Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate o(1/n). *CoRR*, abs/1306.2119, 2013. URL http://arxiv.org/abs/1306.2119.

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 207–216. IEEE, 2013.

Peter L Bartlett and Ambuj Tewari. Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. *arXiv preprint arXiv:1205.2661*, 2012.

David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2010.04.005. URL https://www.sciencedirect.com/science/article/pii/S1063520310000552.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Edoardo Di Napoli, Eric Polizzi, and Yousef Saad. Efficient estimation of eigenvalue counts in an interval. *Numerical Linear Algebra with Applications*, 23(4):674–692, March 2016. doi: 10.1002/nla.2048. URL https://doi.org/10.1002/nla.2048.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *arXiv preprint arXiv:1209.1873*, 2012.

David I. Shuman, Mohammad Javad Faraji, and Pierre Vandergheynst. A multiscale pyramid transform for graph signals. *IEEE Trans. Signal Process.*, 64(8):2119–2134, 2016. doi: 10.1109/TSP.2015.2512529. URL https://doi.org/10.1109/TSP.2015.2512529.