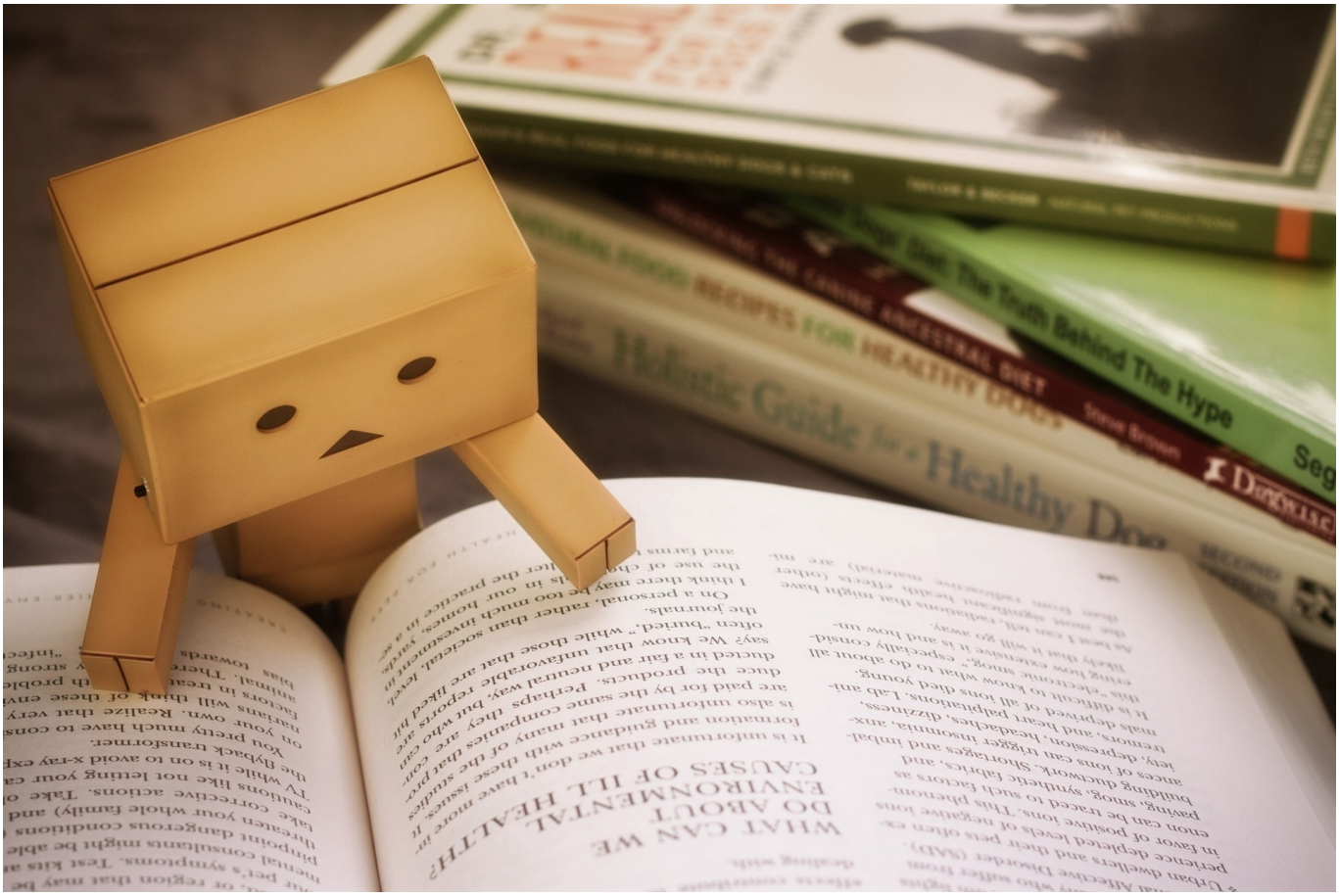
Your Guide to Natural Language Processing (NLP)

How machines process and understand human language

Diego Lopez Yse Follow
Jan 16 · 13 min read



Everything we express (either verbally or in written) carries huge amounts of information. The topic we choose, our tone, our selection of words, everything adds some type of information that can be interpreted and value extracted from it. In theory, we can understand and even predict human behaviour using that information.

But there is a problem: one person may generate hundreds or thousands of words in a declaration, each sentence with its corresponding complexity. If you want to scale and

analyze several hundreds, thousands or millions of people or declarations in a given geography, then the situation is unmanageable.

Data generated from conversations, declarations or even tweets are examples of unstructured data. **Unstructured data** doesn't fit neatly into the traditional row and column structure of relational databases, and represent the vast majority of data available in the actual world. It is messy and hard to manipulate. Nevertheless, thanks to the advances in disciplines like machine learning a big revolution is going on regarding this topic. Nowadays it is no longer about trying to interpret a text or speech based on its keywords (the old fashioned mechanical way), but about understanding the meaning behind those words (the cognitive way). This way it is possible to detect figures of speech like irony, or even perform sentiment analysis.

> Natural Language Processing or NLP is a field of Artificial Intelligence that gives the machines the ability to read, understand and derive meaning from human languages.

It is a discipline that focuses on the interaction between data science and human language, and is scaling to lots of industries. Today NLP is booming thanks to the huge improvements in the access to data and the increase in computational power, which are allowing practitioners to achieve meaningful results in areas like healthcare, media, finance and human resources, among others.

Use Cases of NLP

In simple terms, NLP represents the automatic handling of natural human language like speech or text, and although the concept itself is fascinating, the real value behind this technology comes from the use cases.

NLP can help you with lots of tasks and the fields of application just seem to increase on a daily basis. Let's mention some examples:

- NLP enables the recognition and **prediction of diseases** based on electronic health records and patient's own speech. This capability is being explored in health conditions that go from cardiovascular diseases to depression and even schizophrenia. For example, Amazon Comprehend Medical is a service that uses NLP to extract disease conditions, medications and treatment outcomes from patient notes, clinical trial reports and other electronic health records.

- Organizations can determine what customers are saying about a service or product by identifying and extracting information in sources like social media. This **sentiment analysis** can provide a lot of information about customers choices and their decision drivers.

- An inventor at IBM developed a **cognitive assistant** that works like a personalized search engine by learning all about you and then remind you of a name, a song, or anything you can't remember the moment you need it to.

- Companies like Yahoo and Google filter and classify your emails with NLP by analyzing text in emails that flow through their servers and **stopping spam** before they even enter your inbox.

- To help **identifying fake news**, the NLP Group at MIT developed a new system to determine if a source is accurate or politically biased, detecting if a news source can be trusted or not.

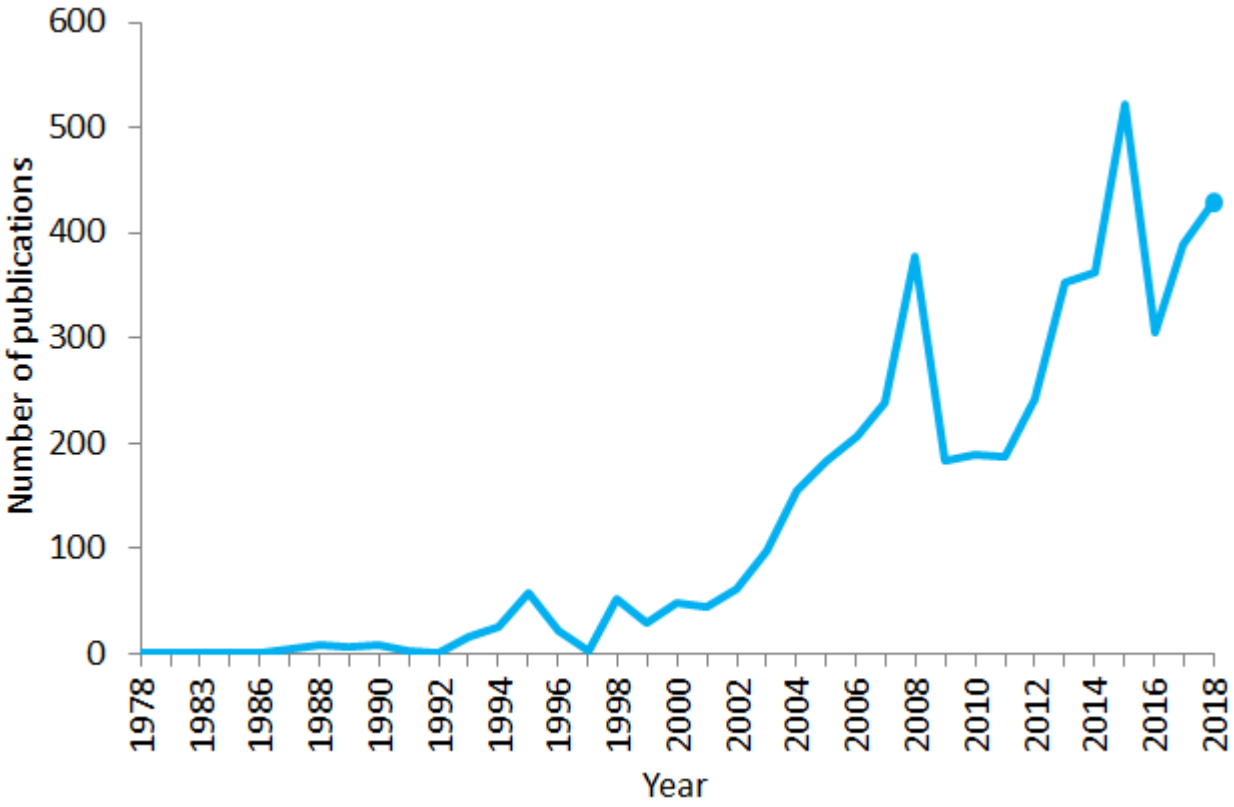
- Amazon's Alexa and Apple's Siri are examples of intelligent **voice driven interfaces**that use NLP to respond to vocal prompts and do everything like find a particular shop, tell us the weather forecast, suggest the best route to the office or turn on the lights at home.

- Having an insight into what is happening and what people are talking about can be very valuable to **financial traders**. NLP is being used to track news, reports, comments about possible mergers between companies, everything can be then incorporated into a trading algorithm to generate massive profits. Remember: buy the rumor, sell the news.

- NLP is also being used in both the search and selection phases of **talent recruitment**, identifying the skills of potential hires and also spotting prospects before they become active on the job market.

- Powered by IBM Watson NLP technology, LegalMation developed a platform to automate routine **litigation tasks** and help legal teams save time, drive down costs and shift strategic focus.

NLP is particularly booming in the **healthcare industry**. This technology is improving care delivery, disease diagnosis and bringing costs down while healthcare organizations are going through a growing adoption of electronic health records. The fact that clinical documentation can be improved means that patients can be better understood and benefited through better healthcare. The goal should be to optimize their experience, and several organizations are already working on this.



Number of publications containing the sentence “natural language processing” in PubMed in the period 1978–2018. As of 2018, PubMed comprised more than 29 million citations for biomedical literature

Companies like Winterlight Labs are making huge improvements in the treatment of Alzheimer’s disease by monitoring cognitive impairment through speech and they can also support clinical trials and studies for a wide range of central nervous system disorders. Following a similar approach, Stanford University developed Woebot,

a **chatbot therapist** with the aim of helping people with anxiety and other disorders.

But serious controversy is around the subject. A couple of years ago Microsoft demonstrated that by analyzing large samples of search engine queries, they could identify internet users who were suffering from pancreatic cancer even before they have received a diagnosis of the disease. How would users react to such diagnosis? And what would happen if you were tested as a false positive? (meaning that you can be diagnosed with the disease even though you don't have it). This recalls the case of Google Flu Trends which in 2009 was announced as being able to predict influenza but later on vanished due to its low accuracy and inability to meet its projected rates.

NLP may be the key to an effective clinical support in the future, but there are still many challenges to face in the short term.

Basic NLP to impress your non-NLP friends

The main drawbacks we face these days with NLP relate to the fact that language is very tricky. The process of understanding and manipulating language is extremely complex, and for this reason it is common to use different techniques to handle different challenges before binding everything together. Programming languages like Python or R are highly used to perform these techniques, but before diving into code lines (that will be the topic of a different article), it's important to understand the concepts beneath them. Let's summarize and explain some of the most frequently used algorithms in NLP when defining the vocabulary of terms:

Bag of Words

Is a commonly used model that allows you to count all words in a piece of text. Basically it creates an occurrence matrix for the sentence or document, disregarding grammar and word order. These word frequencies or occurrences are then used as features for training a classifier.

To bring a short example I took the first sentence of the song "Across the Universe" from The Beatles:

> *Words are flowing out like endless rain into a paper cup,*
>
> *They slither while they pass, they slip away across the universe*

Now let's count the words:

	words	rain	a	paper	they	slip	the	universe	...
Words are flowing out like endless rain into a paper cup,	1	1	1	1	0	0	0	0	...
They slither while they pass, they slip away across the universe	0	0	0	0	3	1	1	1	...

This approach may reflect several downsides like the absence of semantic meaning and context, and the facts that stop words (like "the" or "a") add noise to the analysis and some words are not weighted accordingly ("universe" weights less than the word "they").

To solve this problem, one approach is to rescale the frequency of words by how often they appear in all texts (not just the one we are analyzing) so that the scores for frequent words like "the", that are also frequent across other texts, get penalized. This approach to scoring is called **"Term Frequency — Inverse Document Frequency" (TFIDF)**, and improves the bag of words by weights. Through TFIDF frequent terms in the text are "rewarded" (like the word "they" in our example), but they also get "punished" if those terms are frequent in other texts we include in the algorithm too. On the contrary, this method highlights and "rewards" unique or rare terms considering all texts. Nevertheless, this approach still has no context nor semantics.

Tokenization

Is the process of segmenting running text into sentences and words. In essence, it's the task of cutting a text into pieces called *tokens*, and at the same time throwing away certain characters, such as punctuation. Following our example, the result of tokenization would be:

Pretty simple, right? Well, although it may seem quite basic in this case and also in languages like English that separate words by a blank space (called segmented languages) not all languages behave the same, and if you think about it, blank spaces alone are not sufficient enough even for English to perform proper tokenizations. Splitting on blank spaces may break up what should be considered as one token, as in the case of certain names (e.g. San Francisco or New York) or borrowed foreign phrases (e.g. laissez faire).

Tokenization can remove punctuation too, easing the path to a proper word segmentation but also triggering possible complications. In the case of periods that follow abbreviation (e.g. dr.), the period following that abbreviation should be considered as part of the same token and not be removed.

The tokenization process can be particularly problematic when dealing with biomedical text domains which contain lots of hyphens, parentheses, and other punctuation marks.

For deeper details on tokenization, you can find a great explanation in this article.

Stop Words Removal

Includes getting rid of common language articles, pronouns and prepositions such as “and”, “the” or “to” in English. In this process some very common words that appear to provide little or no value to the NLP objective are filtered and excluded from the text to be processed, hence removing widespread and frequent terms that are not informative about the corresponding text.

Stop words can be safely ignored by carrying out a lookup in a pre-defined list of keywords, freeing up database space and improving processing time.

There is no universal list of stop words. These can be pre-selected or built from scratch. A potential approach is to begin by adopting pre-defined stop words and add words to the list later on. Nevertheless it seems that the general trend over the past

time has been to go from the use of large standard stop word lists to the use of no lists at all.

The thing is stop words removal can wipe out relevant information and modify the context in a given sentence. For example, if we are performing a sentiment analysis we might throw our algorithm off track if we remove a stop word like “not”. Under these conditions, you might select a minimal stop word list and add additional terms depending on your specific objective.

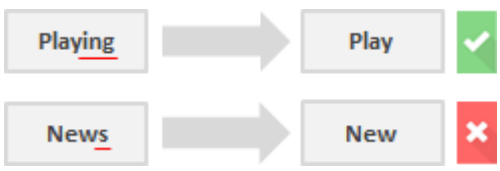
Stemming

Refers to the process of slicing the end or the beginning of words with the intention of removing affixes (lexical additions to the root of the word).

> Affixes that are attached at the beginning of the word are called prefixes (e.g. “astro” in the word “astrobiology”) and the ones attached at the end of the word are called suffixes (e.g. “ful” in the word “helpful”).

The problem is that affixes can create or expand new forms of the same word (called *inflectional* affixes), or even create new words themselves (called *derivational* affixes). In English, prefixes are always derivational (the affix creates a new word as in the example of the prefix “eco” in the word “ecosystem”), but suffixes can be derivational (the affix creates a new word as in the example of the suffix “ist” in the word “guitarist”) or inflectional (the affix creates a new form of word as in the example of the suffix “er” in the word “faster”).

Ok, so how can we tell the difference and chop the right bit?



A possible approach is to consider a list of common affixes and rules (Python and R languages have different libraries containing affixes and methods) and perform stemming based on them, but of course this approach presents limitations. Since stemmers use algorithmics approaches, the result of the stemming process may not be an actual word or even change the word (and sentence) meaning. To offset this effect you can edit those predefined methods by adding or removing affixes and rules, but you must consider that you might be improving the performance in one area while producing a degradation in another one. Always look at the whole picture and test your model's performance.

So if stemming has serious limitations, why do we use it? First of all, it can be used to correct spelling errors from the tokens. **Stemmers are simple to use and run very fast**(they perform simple operations on a string), and if speed and performance are important in the NLP model, then stemming is certainly the way to go. Remember, we use it with the objective of improving our performance, not as a grammar exercise.

Lemmatization

Has the objective of reducing a word to its base form and grouping together different forms of the same word. For example, verbs in past tense are changed into present (e.g. "went" is changed to "go") and synonyms are unified (e.g. "best" is changed to "good"), hence standardizing words with similar meaning to their root. Although it seems closely related to the stemming process, lemmatization uses a different approach to reach the root forms of words.

> *Lemmatization resolves words to their dictionary form (known as* lemma*) for which it requires detailed dictionaries in which the algorithm can look into and link words to their corresponding lemmas.*

For example, the words *"running", "runs"* and *"ran"* are all forms of the word "*run*", so "*run*" is the lemma of all the previous words.

Lemmatization also takes into consideration the context of the word in order to **solve other problems like disambiguation**, which means it can discriminate between identical words that have different meanings depending on the specific context. Think about words like “bat” (which can correspond to the animal or to the metal/wooden club used in baseball) or “bank” (corresponding to the financial institution or to the land alongside a body of water). By providing a part-of-speech parameter to a word (whether it is a noun, a verb, and so on) it’s possible to define a role for that word in the sentence and remove disambiguation.

As you might already pictured, lemmatization is a much more resource-intensive task than performing a stemming process. At the same time, since it requires more knowledge about the language structure than a stemming approach, it **demands more computational power** than setting up or adapting a stemming algorithm.

Topic Modeling

Is as a method for uncovering hidden structures in sets of texts or documents. In essence it clusters texts to discover latent topics based on their contents, processing individual words and assigning them values based on their distribution. This technique is based on the assumptions that each document consists of a mixture of topics and that each topic consists of a set of words, which means that if we can spot these hidden topics we can unlock the meaning of our texts.

From the universe of topic modelling techniques, **Latent Dirichlet Allocation (LDA)** is probably the most commonly used. This relatively new algorithm (invented less than 20 years ago) works as an unsupervised learning method that discovers different topics underlying a collection of documents. In **unsupervised learning** methods like this one, there is no output variable to guide the learning process and data is explored by algorithms to find patterns. To be more specific, LDA finds groups of related words by:

1. Assigning each word to a random topic, where the user defines the number of topics it wishes to uncover. You don’t define the topics themselves (you define just the number of topics) and the algorithm will map all documents to the topics in a way that words in each document are mostly captured by those imaginary topics.

2. The algorithm goes through each word iteratively and reassigns the word to a topic taking into considerations the probability that the word belongs to a topic, and the probability that the document will be generated by a topic. These probabilities are calculated multiple times, until the convergence of the algorithm.

Unlike other clustering algorithms like *K-means* that perform hard clustering (where topics are disjointed), LDA assigns each document to a mixture of topics, which means that each document can be described by one or more topics (e.g. Document 1 is described by 70% of topic A, 20% of topic B and 10% of topic C) and reflect more realistic results.

Topic modeling is extremely useful for classifying texts, building recommender systems (e.g. to recommend you books based on your past readings) or even detecting trends in online publications.

How does the future look like?

At the moment NLP is battling to detect nuances in language meaning, whether due to lack of context, spelling errors or dialectal differences.

On March 2016 Microsoft launched *Tay*, an Artificial Intelligence (AI) chatbot released on Twitter as a NLP experiment. The idea was that as more users conversed with Tay, the smarter it would get. Well, the result was that after 16 hours Tay had to be removed due to its racist and abusive comments:

Microsoft learnt from its own experience and some months later released *Zo*, its second generation English-language chatbot that won't be caught making the same mistakes as its predecessor. Zo uses a combination of innovative approaches to recognize and generate conversation, and other companies are exploring with bots that can remember details specific to an individual conversation.

Although the future looks extremely challenging and full of threats for NLP, the discipline is developing at a very fast pace (probably like never before) and we are likely to reach a level of advancement in the coming years that will make complex applications look possible.

Thanks Jesús del Valle , Jannis Busch and Sabrina Steinert for your valuable inputs

> *Interested in these topics? Follow me on Linkedin or Twitter*

Machine Learning Naturallanguageprocessing NLP Artificial Intelligence

Data Science

Medium About Help Legal