Chapter 3: Natural Language Processing: An Overview

Hessam Amini, Farhood Farahnak and Leila Kosseim

Pages: 35–55

https://doi.org/10.1142/9789811203527_0003

Natural Language Processing: An Overview

Hessam Amini*, Farhood Farahnak* and Leila Kosseim

Computational Linguistics at Concordia (CLaC) Laboratory
Department of Computer Science and Software Engineering
Concordia University, Montreal, Canada H3G 2W1
first.last@concordia.ca

This chapter provides an overview of the field of Natural Language Processing (NLP), a sub-field of Artificial Intelligence (AI) that aims to build automatic systems that can understand or produce texts in natural language. The intended audience is a non-technical reader with no particular background in linguistics or computer science.

The chapter first characterizes natural language and explains why dealing with such unstructured data automatically is a challenge. Examples of typical applications of NLP are then provided ranging from low-level tasks to end-user everyday systems. As much of the work in AI, NLP has gone through three main eras: symbolic approaches, machine learning driven approaches, and more recently, deep learning driven approaches. These three paradigms will be described, with a particular emphasis on the current one, deep learning, which, in only a few years, has led to exciting results and allowed applications, such as conversational agents and machine translation, to become accessible and usable to the public.

1. Introduction

Today, an overwhelming quantity of textual information is available in electronic form. These texts are written in *natural language*, for example, English, French, Spanish, ... but are aimed for human consumption. Developing automatic tools so that machines can understand the content of such documents or produce them automatically is the goal of Natural Language Processing (or NLP). This includes 1) understanding the content of

* Authors, listed in alphabetical order, contributed equally.

documents written in natural language, also known as Natural Language Understanding (or NLU) and 2) generating texts in natural language, also known as Natural Language Generation (or NLG). Practical applications of NLP include systems such as web search engines, text summarizers, word completion, question answering systems, conversational agents, sentiment analyzers, ... to name only a few.

1.1. *Natural versus Artificial Languages*

Natural language refers to the language used for human communication such as English, French, ... As opposed to natural language, *artificial language* refers to a language that was designed. These languages may include human-like languages created for entertainment purposes (such as J.R.R. Tolkien's *Elvish* language[1] or David J. Peterson's *Valyrian* and *Dothraki* languages[2]), for specific practical purposes (such as Esperanto), or for technological and scientific reasons (such as programming languages).

Natural languages have several key differences with artificial languages that make them challenging to process automatically:

Natural languages evolve due to human unconscious and conscious factors, including social, historical, and psychological factors. As natural languages evolve, new words and expressions are created and included in the language. A sentence such as *I forgot my iPhone in your SUV* would have been incomprehensible only a few years ago. NLP systems will therefore need to deal with issues such as an open lexicon, unknown words, ...

Natural languages are robust as they are meant for human communication. Syntactic rules, although typically rather complex, are often flexible. If a semi-colon is required by the grammar, one can easily substitute it for a colon, and the reader will still very likely understand the text. If a word is misspelled, or a word in unknown by the reader, the text will still likely be understandable. NLP systems therefore need to address issues such as informally written texts, misspelled words, ungrammatical sentences, ...

Natural languages are ambiguous at many levels. For example, a single word may have different meanings (e.g. a *chair* may be a person or a piece of furniture), a sentence may have different syntactic parses (e.g. in the sentence *The man saw Jane with the telescope*, the phrase

with the telescope can qualify either the verb *saw* or the noun *Jane*), or a sentence may have different interpretations depending on its context. These ambiguities create challenges for NLP systems. For example, a human will easily understand that in the sentence *the chair cancelled the meeting*, the word *chair* refers to a person, but automatically ruling out the sense of furniture will need to be addressed.

1.2. *Natural Language Understanding versus Generation*

The field of NLP is composed of two sub-fields:

(1) **Natural Language Understanding (NLU)** takes texts as input. NLU tries to develop techniques to understand and interpret texts in natural language in order to perform a variety of decisions: act according to some instructions, extract relevant information from a text, provide an answer to a question, classify or summarize a document, or create an intermediary representation of the text that can be used for further processing in other applications.

(2) **Natural Language Generation (NLG)** produces texts as output. As opposed to NLU, NLG takes as input some representation of the content to communicate (which can be of various forms[3]) and tries to generate as output a human-like textual data such as an answer, an article, an email...

The ultimate goal of NLU is to understand a text as well as a human would; while the ultimate goal of NLG is to produce a text as well as a human would. These two tasks can be seen in action in conversational agents, where the system must understand the user's sentences (NLU) in order to respond accordingly (NLG).

1.3. *Components of Natural Language Processing Systems*

In order to process natural language automatically, systems are traditionally divided into different components, each of which deals with a different aspect of language. Following the field of linguistics, it is standard to consider the following components:

Lexical Analysis studies individual words. Automatically, converting a sequence of characters into a sequence of words and sentences is the task

of lexical analysis. This includes tasks such as tokenization (e.g. *I don't* → *I / don't*), lemmatization (e.g. *I / do / not*), sentence boundary detection, ...

Syntax refers to the set of rules that govern how words can be arranged together in order to form grammatical sentences. For example, in English, the sentence *I like to read* is grammatical, but the same set of words in a different order, for example, *like I read to*, is not. Understanding the underlying structure of a sentence gives very strong cues about its meaning.

Semantics refers to the meaning of words, phrases and sentences and how these meanings are related. *Lexical semantics* studies the meanings of individual words and the semantic relations that they have with other words. For example, lexical semantics is involved when determining that the word *chair* in the sentence *The department voted for Jane to be the new chair* refers to a person rather than a piece of furniture. On the other hand, *compositional semantics* studies how the meaning of larger phrases and sentences can be composed from the meaning of their individual words and their relationships with each other.

Discourse goes beyond individual sentences and tries to capture the relations between sentences in order to understand the text as a whole. This includes the study of referring expressions (e.g. the use of pronouns to refer to entities already mentioned in the text) and discourse relations that indicate the logical relation between textual elements. For example in *Jack is hungry, because he did not eat*, it is important to identify that there is a causality relation between the two clauses.

Pragmatics tries to go beyond the literal meaning and studies how the context influences the meaning of a text. The context can be linguistic (e.g. the other words around), cultural, or situtational (e.g. in which situation the text is written/said). For example, if on the street, someone asks a passer-by *Do you have some change?*, the questioner does not really want to be informed if the other has or does not have change; but rather is requesting to be given some change.

World Knowledge takes into account the assumptions or background information about the world (such as history, facts, ...) to truly understand

the text. For example, in the sentence *The trophy would not fit in the brown suitcase because it was too big*,[4] a human would easily understand that the pronoun *it* refers to the *trophy*; whereas in *The trophy would not fit in the brown suitcase because it was too small*, the pronoun *it* refers to the *suitcase*.

This knowledge, referred to as *world knowledge* or *common sense knowledge*, is naturally acquired by humans through their living experience. Building systems that understand or produce natural language as well as a human requires to take into account world knowledge.

2. Applications

Let us now explore examples of successful applications of NLP. We categorized these applications into two classes: low-level and high-level applications.

2.1. *Low-Level Applications*

Low-level applications of NLP are typically not used by end-users, but rather compute core linguistic representations that are used in order to develop or improve high-level applications that are used by end-users. Standard low-level applications include:

Part-of-Speech (PoS) Tagging. Part-of-Speech tags are grammatical labels that are assigned to words in a particular sentence. Such tags can be general, such as *Noun*, *Verb*, *Adjective*, or more fine-grained, such as *Noun-Singular*, *Verb-be-3rd-Person-Singular-Present*, ...

Over the years, a multitude of automatic PoS taggers have been developed following a variety of approaches (see Section 3). Today PoS taggers achieve around 97% accuracy for English.[5] However, PoS tagging is not a solved problem, as these tools are often domain-dependent: a PoS tagger trained on a specific domain (e.g. biology) does not perform as well on another domain (e.g. computer science).

Named Entity Recognition (NER) is concerned with finding and labelling a predefined set of semantic expressions, such as PERSONS, PLACES, ORGANIZATIONS, EXPRESSIONS OF TIME, etc. For example, given the sentence *The meeting with Jane Young, the CEO of ABC inc., was July 3 2017*, an NER system will determine that *Jane Young* is a PERSON, *ABC inc.* is an ORGANIZATION, and *July 3 2017* is a DATE.

Extracting named entities is seen as a segmentation problem and is further complicated by embedded entities. For example, *University of Montreal* is a named entity, but *Montreal* on its own is also an entity. Labelling named entities is also challenging, since the same entity may be labeled differently depending on the context. For example, *Louis Vuitton* can be a PERSON, an ORGANIZATION or a COMMERCIAL PRODUCT. State-of-the-art NER systems achieve a performance of around 90% for English, while for other languages, the performance is significantly lower.[6]

Co-reference Resolution and Generation. In a text, several words or phrases can refer to the same entity. For example, in the sentence *I saw Alex when he was going home*, both *Alex* and *he* refer to the same entity. Finding the referent of expressions is essential to understand a text.

Co-reference resolution aims to identify these co-references automatically. State-of-the-art systems have achieved performances of around 65% to 74% for English,[7] which shows that much work still needs to be done in this field. As opposed to co-reference resolution, co-reference generation (often referred to as *referring expressions generation*) aims to find the most natural words or phrases that an NLG system should use to refer to an entity.[8]

Word Sense Disambiguation. As indicated in Section 1.1, natural language is ambiguous. In particular, one word or expression may have more than one meaning; for example, the word *bass* may refer to a fish or a man with a low-pitched singing voice. Word Sense Disambiguation (WSD) is the task of identifying the sense of a word in its context. For example in *the bass was excellent on the grill* a WSD system will find that most likely *bass* refers to a fish.

Similarly to PoS tagging, WSD can be considered as a classification problem; however, different linguistic features are used. In WSD, the words immediately surrounding the ambiguous word give very strong clues about its meaning. State-of-the-art systems in WSD achieve performances around 95%.[9]

Syntactic Parsing determines how words in a sentence are grouped into constituents following a specific grammar. The result is one, or more often, multiple, *parse trees*. Finding the proper parse tree of a sentence helps determine what the sentence means, and hence is very useful for applications

such as question answering. For example, given the sentence *The man saw Jane with the telescope*, two parse trees are shown in Figure 1.

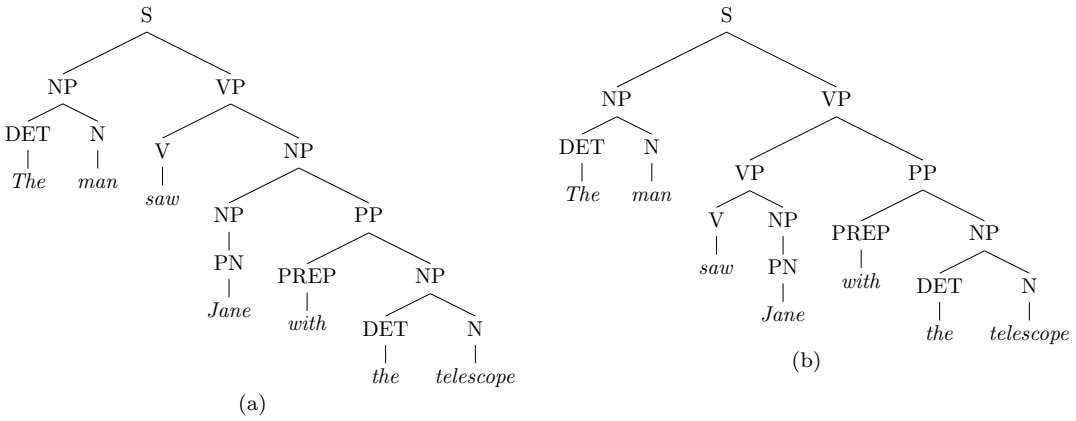

Fig. 1. Two possible parse trees for the sentence *The man saw Jane with the telescope.*

In Figure 1(a), the constituent *with the telescope* is part of the noun phrase (NP) – in this case, the sentence means that Jane was carrying the telescope; whereas in Figure 1(b), *with the telescope* is part of the verb phrase (VP) – in this case, the sentence means that the telescope was the instrument used to see Jane.

Today, syntactic parsers can reach performances that are comparable to human experts on English texts (with performances near 92%);[10,11] however much work still needs to be done to improve their performance on informal texts, such as tweets, or languages other than English.[12]

Paraphrase Detection & Generation. A paraphrase is a text that states the same meaning as another, but using different words or grammatical structures. For example, the two sentences *Scientists studied this case* and *This case was studied by scientists* are paraphrases.

The automatic detection of paraphrases is crucial for several applications, such as text summarization, plagiarism detection, and question answering. Today, state-of-the-art paraphrase detection systems perform very well on well-written English texts;[13] however, much work still needs to be done to improve their performance on informal texts, such as tweets.[14]

On the other hand, paraphrase generation tries to generate paraphrases automatically. This task is particularly useful for conversational agents in

order to vary the output of the system. Recent advancements in deep learning for NLP, in particular sequence-to-sequence models (see Section 3.3) have led to the development of systems that generate paraphrases of very high quality.[14]

2.2. *High-Level Applications.*

As opposed to low-level applications, high-level applications aim to solve problems for the end-user. Typical examples include:

Information Extraction (IE) tries to extract specific structured or semi-structured information from texts.[15] For example, from the sentence *Jane's son, Jim, works at Microsoft*, an IE system would exact the relations CHILD-OF(*Jane*, *Jim*) and EMPLOYEE-OF(*Microsoft*, *Jim*).

IE systems typically first use NER and co-reference resolution (see Section 2.1) to identify named entities (*Jane*, *Jim* and *Microsoft*). Then the semantic relations between these entities are found. These relations can be very general, such as CHILD-OF, EMPLOYEE-OF, ... or specific to a particular domain such as MUTATION-EXTRACTION, MUTATION-TO-GENE-ASSOCIATION, ... in biomedical applications.[16] Finally, many IE systems organize the extracted information by automatically filling slots in predefined templates.

Sentiment Analysis automatically detects affective states in a text. The most basic task in sentiment analysis is to identify the polarity of a text – such as positive, negative, or neutral. In a more complex setting, sentiment analysis can extract the emotional state (e.g. happy, sad, angry, ...) regarding a specific aspect or feature of an entity.

Sentiment analysis has a wide range of applications in marketing and customer service, as it allows to mine customer reviews. Other useful applications include recommender systems, where sentiment analysis is used to understand the preferences of online customers or social media users in order to provide them with appropriate advertisements or recommendations.

Summarization is the task of automatically creating a summary from a natural language text that includes the most important points of the text. Two main approaches are typically followed: *extraction-based summarization* and *abstraction-based approach*. Extractive summarisation creates a summary by identifying and extracting key parts of the original text. On

the other hand, abstractive summarization first tries to represent the content of the original text, and then produces a summary using NLG techniques.

Question Answering (QA) tries to automatically provide a specific answer to questions that are asked in natural language. For example, given the question *Where is the Eiffel Tower*, a QA system will search through a document collection and extract the answer *Paris*. QA can either be closed-domain or open-domain. Closed-domain QA systems focus on answering questions in specific domains (e.g. law, medicine, ...), while open-domain QA systems are capable of answering more general questions on any domain.

Conversational Agents (also known as *dialogue systems*) are applications that are able to hold coherent conversations with humans. Conversational agents should be capable of understanding the user's intentions and responding appropriately and fluently.

Two types of conversational agents have been developed: *task-oriented bots* and *chatbots*. Task-oriented bots are aimed at performing a specific task, such as placing an order, booking a hotel/flight, or scheduling an event, via a conversation. On the other hand, chatbots are mostly designed for entertainment purposes. Today, thanks to advances in deep learning for NLP (see Section 3.3), the performance of conversational agents has reached a level that allows them to be used in our daily life for simple tasks.

3. NLP Techniques

NLP is a sub-field of Artificial Intelligence (AI). As such, it followed the same technical trends of the field of AI. In this section, we will describe the three main approaches used in NLP over the years: *symbolic NLP*, *statistical NLP*, and *deep learning for NLP*.

3.1. *Symbolic NLP*

The field of NLP dates back to the early years of Artificial Intelligence (AI) in the 1950's, when Alan Turing published the seminal paper *Computing Machinery and Intelligence*[17] in which he directly relates machine intelligence to the ability to communicate with humans through natural language.

From the 1950's to the 1990's, NLP was performed by experts versed in both linguistics and computer science. Following the trend in AI in those days, NLP used a rule-based, prescriptive approach, where experts developed rules by hand to describe how language ought to be. These techniques were also known as *knowledge intensive* approaches, because experts would manually encode their knowledge into symbolic logical rules in order to perform NLP tasks. Developing hand-crafted rule-based systems was time consuming and expensive. Prototype systems were developed, but very little high-level applications made their way to the end-users. On the other hand, many theoretical and fundamental issues were addressed.[18] For example, the work of Noam Chomsky on the structure of language[19] set the groundwork for much advancement in NLP in these days.

3.2. *Statistical NLP*

In the mid-1990s, the field went through its first major paradigm shift. Large annotated corpora became available, and statistical methods and machine learning became more and more attractive to describe how language was actually used in practice. The era of statistical NLP was born.[20] Machine learning techniques were applied to large document collections to automatically identify discriminating linguistic features that were useful for the specific NLP applications to develop. The need for linguistic expertise shifted from writing rules by hand to describing useful linguistic features, and the machine learning algorithms would develop the rules automatically. The use of machine learning was much less time-consuming and expensive, and the heavy use of large corpora allowed the development of more robust low-level as well as high-level NLP applications such as part-of-speech taggers,[21,22] syntactic parsers,[23,24] and named entity taggers.[25,26]

3.3. *Deep Learning for NLP*

Around 2010, the field of NLP, and AI in general, went through another major parading shift: the era of deep learning. The resurgence of neural networks led researchers to push the boundaries even further and eliminate the need to develop hand-crafted features for the machine learning algorithms. The emergence of deep models and end-to-end learning algorithms led to NLP techniques that automatically learn a representation of useful linguistic features. The field went from writing rules by hand, to learning rules automatically (but still hand-crafting linguistic features), to automating the entire process: automating the rules and finding the lin-

guistic features (also known as representations) automatically. In only a few years, deep learning methods have led to impressive improvements to most NLP tasks and have led to the development of many end-user applications such as conversational systems, question answering, more accurate machine translation systems, ...[27–29]

Due to the success of this new paradigm, we will describe these techniques in more detail in the next sections.

3.3.1. *Language Models*

The first successful and wide use of deep learning for NLP was is the area of *Language Modelling*. A Language Model (LM) is used to compute the probability of a sentence being used in a specific language. For example, an LM could determine that the probability of the sequence of words (or sentence) *I will find out* in English is 0.003, but the probability of *I will fine out* is much lower (e.g. 0.0001).

LMs are used in many NLP applications such as machine translation,[30] speech recognition,[31] ... Hence, being able to better estimate the probability of a sentence through a more accurate LM, often directly leads to improved results in many end-user applications.

From the 1990's to 2010, most LM methods were based on a method called *ngram modelling*[20,32] where a large collection of texts was used to count how many times a word follows a specific sequence of previous words.

Early efforts to use neural networks to create LMs date back to the early 1990's (for example[33–35]); but it was only in 2010 that significant improvements over the standard n-gram method was achieved through the use of Recurrent Neural Networks (RNNs).[36] An RNN is a special type of neural network particularly appropriate to deal with sequences, such as sequences of words or sentences. With the concurrent success of deep learning in other domains (such as speech recognition[31] and image classification[37]) researchers started to investigate the use of these models for a variety of NLP tasks such as machine translation,[38,39] text analysis,[6] conversational agents[40,41] and image captioning.[42]

3.3.2. *Language Representation*

Before major breakthroughs could be achieved by deep learning in NLP, another important issue had to be addressed: neural networks require as input a vector of numbers that does not lend itself well to sequences of

words. In order to make better use of neural networks for NLP, an effective way to represent natural language through numbers was needed.

Word Embeddings

Traditionally, to be fed to a neural network, a word was represented as a *one-hot vector*. In such representation, a word is represented as a binary vector of the size of the vocabulary. Only one position in this vector has the value of 1, all others are 0. For example, the one-hot representation of the word *python* may be [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...] where the length of the vector is in the order of a few thousand elements. The problem with this representation is that similar words such as *python* and *boa* do not have similar vector representations. The distance between all one-hot vectors is identical. Hence a neural network will learn independent regularities of the language if the words *python* and *boa* are used as input without generalizing what it has learned to *snakes* or *reptiles*.

What was needed is a vector representation where similar words such as *python* and *boa* have similar representations and dissimilar words such as *python* and *orange* have dissimilar representations. In addition, by reducing the size of the vectors and allowing decimal values instead of binary values, we create a dense vector and reduce sparsity. Embeddings are an alternative to one-hot vectors where each word is represented by such a dense vector. For example using a word embedding, *python* could be represented by [0.34, 0.67, 0.04, 0.06, ...] and the size of the vector is reduced from a few thousand to a few hundred elements. Several methods have been developed to find the values of these embeddings.[43,44] Typically, these methods follow the *distributional hypothesis*, which states that words with similar meanings tend to occur in similar contexts. For example, Word2vec[43] learns the embedding of a word by learning to predict the current word in a text given its surrounding words or vice-versa. Using such embeddings, words with similar characteristics tend to have similar vector representations. In addition, a variety of relations between words seemed to be learned as well. The now famous example of subtracting the vector for *man* from the vector for *king* and adding the vector for *woman* results in a vector that is very close to the vector for the word *queen*: VEC("KING") - VEC("MAN") + VEC("WOMAN") ≈ VEC("QUEEN").[43] This captures the gender relation. Another example is VEC("PARIS") - VEC("FRANCE") + VEC("POLAND") ≈ VEC("WARSAW") which describes the concept of capital city.

Character Embeddings

Instead of using embeddings for words, another level of granularity often used is the character level. In that case, the size of the dictionary is drastically reduced to the size of the alphabet plus a few special characters, such as punctuation marks. This significantly reduces the computation and memory requirements of word-based models that have to deal with large dictionaries.

Although working at the character level may not seen intuitive on a linguistic point of view, it does alleviate the problem of out-of-vocabulary words.[45,46] In addition a suitable character-level language model can to capture the meaning of some unseen words. For example the fact that the character *s* is often used to indicate the plural form.

Sentence Embeddings

Another possibility is to represent an entire sentence as a vector through sentence embeddings. The difficulty in creating such embeddings is that large amounts of data are required to generalize over all possible sentences.[47,48]

Through embeddings (word, character or sentence based) natural language sentences could be more efficiently processed by neural networks, which made these more and more used for NLP applications and several novel neural architectures then followed.

3.3.3. *Model Architectures*

Once the problem of language representation was solved, and the use of neural networks was wide-spread in NLP, another important problem had to be addressed: natural language exhibits was is called *long distance dependencies* that standard neural network models (including RNNs) do not handle well. For example, in the sentence: *My brother, the math teacher who loves fast cars and travelling, biked to work this morning*, a model should be able to "remember" the information *teacher* to understand where *work* refers to, even if the two words are far away from each other in the text. To alleviate this problem, several new types of neural networks had to be developed.

LSTMs and GRUs

Deep networks, including RNNs suffer from the so-called *vanishing* and

exploding gradient problems which stops the network from learning when processing long sentences.[49] One effective way to handle this, is to use LSTMs (Long Short-Term Memory[50]) or their variant, GRUs (Gated Recurrent Unit[39]). These cells try to control the flow of information using *gates* so that important information from the text can be remembered and others can be forgotten. Today, LSTMs and GRUs are the most common choices in most works in NLP.[27]

Sequence-to-Sequence Models

With the growing computational power and availability of larger datasets, more complex neural network models can be used and even combined as an ensemble. Sequence-to-sequence models follow such an approach. A sequence-to-sequence model consists of two RNNs: the first one, called the *encoder*, reads an input sentence word by word and maps it to a single vector representation. The second RNN, called the *decoder*, consists of a language model which generates an output word by word, conditioned on the encoded vector. Hence the encoder plays the role of an NLU system and the decoder performs NLG.

Attention Mechanisms

Although LSTMs and GRUs can deal better with long sentences than standard RNNs, they cannot deal with sentences of arbitrary length. For example, in neural machine translation, the performance of a simple encoder-decoder architecture starts to drop for sentences longer than 20 words.[51,52] To address this problem, *attention mechanisms* were developed, where the decoder generates an output word, not only based on the last encoder output and the previously generated words, but also based on all incremental encoder outputs. This way, the decoder has access to all embedded information for each incremental result of the encoder.

Memory Networks

Although applying attention improves the performance of many NLP tasks[53–55] and allows to deal with long sentences, the model needs more computation and memory as the length of the sentence grows. In a task where a long document should be used as input to the network, the use of an attention mechanism requires to keep track and perform computations for a large number of intermediary vectors, which is highly inefficient. One way to deal efficiently with long documents is to augment the model with an external memory[56] and instead of keeping track of all the words in the

document, the model only keeps track of the information that might be useful later. Choosing what kind of information needs to be written in the memory can itself be learned using another neural network. Memory-augmented models have achieved impressive results in different tasks such as question answering,[57] sentiment analysis[58] and machine translation.[59]

3.3.4. *Successful Applications*

Deep learning techniques have had a great impact on the field of NLP. Not only have they significantly improved the state of the art in many standard NLP tasks, such as machine translation,[60] question answering[57] and conversational agents[41] (see Section 2), but also they have been applied to new NLP applications such as visual question answering[61] and image captioning.[42] Three notable applications are described below.

Machine Translation is one of the most successful applications of deep learning for NLP. Work in this area has led to a new generation of systems called Neural Machine Translation (NMT).[38,39] NMT is based on a sequence-to-sequence model (see Section 3.3.3) where the encoder reads a sentence in the source language and maps it to a context vector (a representation of the source sentence), and the decoder maps the context vector to its translation in the target language. Attention mechanisms (see Section 3.3.3) are typically used to deal with long sentences.[51,62] The amazing successes of NMT allowed the development of end-user applications such as Google's online machine translation.[60]

Image Captioning is a novel application based on the idea of translation via an encoder-decoder; however, instead of translating from a natural language to another, image captioning translates an image to an English sentence that describes it. An encoder receives an image as input and creates a vector representation for it, and then a decoder takes the vector as input and generates a sentence. This idea has led to impressive novel systems.[42]

Conversational Agents described in Section 2.2, constitute one of the earliest applications of AI.[17] Whereas early chatbots were based on handwritten rules (e.g. Eliza[63]), modern conversational agents such as Google Now or Alexa are based on deep learning techniques. Typically, one or two encoders encode the input sentences and their context into a vector

representation, and then a decoder uses this representation to generate a response word by word.[40,41] Although the performance of these new conversational agents is significantly better than their ancestors, they still suffer from several important issues. In particular, these systems tend to generate short and generic responses such as *yes* or *I don't know*.[64] In addition, as these systems do not ground their "understanding" to the real world, they may produce answers that make no sense or are inconsistent. Hence we need to find a appropriate way to take into account pragmatics, discourse information and world-knowledge (see Section 1.3) to generate more natural answers.

4. Conclusion

This chapter has provided a non-technical overview of the field of Natural Language Processing (NLP): one of the earliest sub-fields of Artificial Intelligence (AI) that brings together experts in both computer science and linguistics. We have characterized the complexity of natural language that make its automatic processing much more complex than when dealing with artificial languages. Ambiguity at the lexical, syntactic and semantic levels, as well as the inherent necessity to deal with pragmatic and world-knowledge makes this field of study both challenging and fascinating at the same time.

As much of the work in AI, NLP has gone through three main eras. In the early days (from the 1950's to the 1990's), symbolic computation was the driving paradigm. Hand-written rules developed by linguists and computer scientists drove NLP algorithms. Most systems were developed only at the prototype level as they did not scale well to real-life applications. In the 1990's, as more and more electronic documents became available, more robust statistical and machine learning methods became the norm to mine these large quantities of text collections automatically. Human expertise shifted from writing rules, to hand-crafting linguistic features that were then used by the machine learning methods to discover the rules automatically. Many low-level applications were developed during these days, but only a few made their way to every-day applications. Around 2010, with the advancements of deep learning models, a new era in NLP development was born. A variety of neural-network based approaches were developed to deal specifically with natural language and significantly improved the state of the art in many NLP tasks. Expertise in NLP shifted again from hand-crafting linguistic features, to providing no hand-crafted information

at all. The networks discover automatically both the linguistic features and the rules. With the ever increasing availability of data, and new neural-based architectures, such as RNN's, sequence-to-sequence models, ... new applications such as efficient machine translation and conversational agents finally made their way to end-users.

It is an exciting time to work on natural language processing, and future applications are limitless. The holy grail of building automatic systems that can produce or analyze texts as a human would seem more and more reachable today.

References

1. J. R. R. Tolkien, *The Lord of the Rings*. Allen Unwin (1954).
2. D. J. Peterson, The languages of ice and fire, *Mastering the Game of Thrones: Essays on George RR Martin's A Song of Ice and Fire*. pp. 15–34 (2015).
3. E. Reiter and R. Dale, *Building Natural Language Generation Systems*. Cambridge University Press, New York, USA (2000).
4. H. J. Levesque, E. Davis, and L. Morgenstern. The Winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, pp. 552–561, Rome, Italy (June, 2011).
5. C. D. Manning. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *12th International Conference on Intelligent Text Processing and Computational Linguistics* .
6. G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAALC-HTL 2016, San Diego, California (June, 2016).
7. K. Clark and C. D. Manning. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, ACL 2016, Berlin, Germany (August, 2016).
8. E. Krahmer and K. van Deemter. Graphs and booleans: on the generation of referring expressions. In *Computing Meaning: Volume 3 (Studies in Linguistics and Philosophy)*. Springer (2006).
9. A. Moro, A. Raganato, and R. Navigli, Entity linking meets word sense disambiguation: A unified approach. **2**, 231–244 (05, 2014).
10. D. Chen and C. Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2014, pp. 740–750, Doha, Qatar (October, 2014).
11. O. Vinyals, L. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton. Grammar as a foreign language. In *Proceedings of the 28th International Confer-*

ence on Neural Information Processing Systems - Volume 2, NIPS 2015, pp. 2773–2781, Montreal, Canada (December, 2015).
12. C. Gómez-Rodríguez, I. Alonso-Alonso, and D. Vilares, How important is syntactic parsing accuracy? An empirical evaluation on rule-based sentiment analysis, *Artificial Intelligence Review* (October, 2017).
13. B. Dolan, C. Quirk, and C. Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING 2004, Geneva, Switzerland (2004).
14. B. Agarwal, H. Ramampiaro, H. Langseth, and M. Ruocco, A deep network model for paraphrase detection in short text messages, *Information Processing Management*. **54**(6), 922 – 937 (2018).
15. J. Piskorski and R. Yangarber, *Information Extraction: Past, Present and Future*, In eds. T. Poibeau, H. Saggion, J. Piskorski, and R. Yangarber, *Multi-source, Multilingual Information Extraction and Summarization*, pp. 23–49. Springer, Berlin, Heidelberg (2013).
16. A. S. M. A. Mahmood, T.-J. Wu, R. Mazumder, and K. Vijay-Shanker, Dimex: A text mining system for mutation-disease association extraction, *PLOS ONE*. **11**, 1–26 (04, 2016).
17. A. M. Turing, Computing machinery and intelligence, *Mind*. **59**(236), 433–460 (1950).
18. J. Allen, *Natural Language Understanding (2nd Ed.)*. Benjamin-Cummings Publishing Co., Inc., Redwood City, USA (1995).
19. N. Chomsky, *Syntactic Structures*. The Hague: Mouton (1957).
20. C. D. Manning, C. D. Manning, and H. Schütze, *Foundations of statistical natural language processing*. MIT press (1999).
21. T. Brants. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC 2000, pp. 224–231, Seattle, USA (May, 2000).
22. K. Toutanova and C. D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP 2000, pp. 63–70, Hong Kong (2000).
23. A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Second Conference on Empirical Methods in Natural Language Processing*, EMNLP 1997, pp. 1–10, Providence, USA (August, 1997).
24. S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. A novel use of statistical parsing to extract information from text. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, pp. 226–233, Seattle, USA (May, 2000).
25. A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *In Proceedings of the Sixth Workshop on Very Large Corpora*, pp. 152–160, Montreal, Canada (August, 1998).

26. R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL 2003, pp. 168–171, Edmonton, Canada (June, 2003).
27. T. Young, D. Hazarika, S. Poria, and E. Cambria, Recent trends in deep learning based natural language processing, *arXiv preprint arXiv:1708.02709* (2017).
28. Y. Goldberg and G. Hirst, *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers (2017).
29. L. Deng and Y. Liu, *Deep Learning in Natural Language Processing*. Springer, Singapore (2018).
30. A. Vaswani, Y. Zhao, V. Fossum, and D. Chiang. Decoding with large-scale neural language models improves translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2013, pp. 1387–1392 (2013).
31. G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal Processing Magazine*. **29**(6), 82–97 (2012).
32. D. Jurafsky and J. H. Martin, *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, USA (2009).
33. J. L. Elman, Finding structure in time, *Cognitive Science*. **14**(2), 179–211 (1990).
34. R. Miikkulainen and M. G. Dyer, Natural language processing with modular PDP networks and distributed lexicon, *Cognitive Science*. **15**(3), 343–399 (1991).
35. Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, A neural probabilistic language model, *Journal of Machine Learning Research*. **3**, 1137–1155 (Mar., 2003).
36. T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur. Recurrent neural network based language model. In *Proceedings of the Eleventh Annual Conference of the International Speech Communication Association*, pp. 1045–1048 (01, 2010).
37. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS 2012, pp. 1097–1105, USA (2012).
38. I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS 2014, pp. 3104–3112 (2014).
39. K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2014, pp. 1724–1734, Doha, Qatar (October, 2014).

40. O. Vinyals and Q. V. Le, A neural conversational model, *CoRR* (2015). URL `http://arxiv.org/abs/1506.05869`.
41. I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, Hierarchical neural network generative models for movie dialogues, *CoRR, abs/1507.04808* (2015).
42. O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, Show and tell: A neural image caption generator, *2015 IEEE Conference on Computer Vision and Pattern Recognition*. pp. 3156–3164 (2015).
43. T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, Efficient estimation of word representations in vector space, *CoRR* (2013).
44. J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2014, pp. 1532–1543, Doha, Qatar (October, 2014).
45. A character-level decoder without explicit segmentation for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, series = ACL 2016, author = Chung, Junyoung and Cho, Kyunghyun and Bengio, Yoshua, month = aug, year = 2016, pages = 1693–1703*, Berlin, Germany .
46. Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI 2016, pp. 2741–2749 (2016).
47. D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al., Universal sentence encoder, *arXiv preprint arXiv:1803.11175* (03, 2018).
48. S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *International Conference on Learning Representations* (2018).
49. Y. Bengio, P. Simard, and P. Frasconi, Learning long-term dependencies with gradient descent is difficult, *IEEE Transactions on Neural Networks*. **5**(2), 157–166 (1994).
50. S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural computation*. **9**(8), 1735–1780 (1997).
51. D. Bahdanau, K. Cho, and Y. Bengio, Neural machine translation by jointly learning to align and translate, *CoRR* (2014). URL `http://arxiv.org/abs/1409.0473`.
52. M.-T. Luong and C. D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2016, pp. 1054–1063, Berlin, Germany (August, 2016).
53. O. Vinyals, L. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton. Grammar as a foreign language. In eds. C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, *Advances in Neural Information Processing Systems 28*, NIPS 2015, pp. 2773–2781 (2015).
54. R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. In *Proceedings of the International Conference on Learning*

Representations (2018).
55. K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pp. 1693–1701, MIT Press, Cambridge, MA, USA (2015).
56. A. Graves, G. Wayne, and I. Danihelka, Neural turing machines, *CoRR* (2014). URL `http://arxiv.org/abs/1410.5401`.
57. S. Sukhbaatar, a. szlam, J. Weston, and R. Fergus. End-to-end memory networks. In eds. C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, *Advances in Neural Information Processing Systems 28*, NIPS 2015, pp. 2440–2448 (2015).
58. P. Chen, Z. Sun, L. Bing, and W. Yang. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2017, pp. 452–461 (2017).
59. Y. Feng, S. Zhang, A. Zhang, D. Wang, and A. Abel. Memory-augmented neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2017, pp. 1390–1399 (2017).
60. Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. S. Corrado, M. Hughes, and J. Dean, Google's neural machine translation system: Bridging the gap between human and machine translation, *CoRR* (2016).
61. S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. In *International Conference on Computer Vision*, ICCV 2015 (2015).
62. T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2015, pp. 1412–1421, Lisbon, Portugal (September, 2015).
63. J. Weizenbaum, Eliza—a computer program for the study of natural language communication between man and machine, *Communications of the ACM*. **9**(1), 36–45 (Jan., 1966).
64. J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT 2016, pp. 110–119 (2016).