Natural Language Processing Advancements By Deep Learning: A Survey

Amirsina Torfi, *Member, IEEE,* Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A. Fox, *Fellow, IEEE*

***Abstract*—Natural Language Processing (NLP) helps empower intelligent machines by enhancing a better understanding of the human language for linguistic-based human-computer communication. Recent developments in computational power and the advent of large amounts of linguistic data have heightened the need and demand for automating semantic analysis using data-driven approaches. The utilization of data-driven strategies is pervasive now due to the significant improvements demonstrated through the usage of deep learning methods in areas such as Computer Vision, Automatic Speech Recognition, and in particular, NLP. This survey categorizes and addresses the different aspects and applications of NLP that have benefited from deep learning. It covers core NLP tasks and applications, and describes how deep learning methods and models advance these areas. We further analyze and compare different approaches and state-of-the-art models.**

***Index Terms*—Natural Language Processing, Deep Learning, Artificial Intelligence**

I. Introduction

NATURAL Language Processing (NLP) is a sub-discipline of computer science providing a bridge between natural languages and computers. It helps empower machines to understand, process, and analyze human language [1]. NLP's significance as a tool aiding comprehension of human-generated data is a logical consequence of the context-dependency of data. Data becomes more meaningful through a deeper understanding of its context, which in turn facilitates text analysis and mining. NLP enables this with the communication structures and patterns of humans.

Development of NLP methods is increasingly reliant on data-driven approaches which help with building more powerful and robust models [2], [3]. Recent advances in computational power, as well as greater availability of big data, enable deep learning, one of the most appealing approaches in the NLP domain [2]–[4], especially given that deep learning has already demonstrated superior performance in adjoining fields like Computer Vision [5]–[7] and Speech Recognition [8], [9]. These developments led to a paradigm shift from traditional to novel data-driven approaches aimed at advancing NLP. The reason behind this shift was simple: new approaches are more promising regarding results, and are easier to engineer.

Amirsina Torfi, Yaser Keneshloo, and Edward A. Fox were with the Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA, 24060 USA e-mail: (amirsina.torfi@gmail.com, yaserkl@vt.edu, fox@vt.edu). Rouzbeh A. Shirvani is an independent researcher, e-mail: (rouzbeh.asghari@gmail.com). Nader Tavaf was with the University of Minnesota Twin Cities, Minneapolis, MN, 55455 USA e-mail: (tavaf001@umn.edu).

As a sequitur to remarkable progress achieved in adjacent disciplines utilizing deep learning methods, deep neural networks have been applied to various NLP tasks, including part-of-speech tagging [10]–[12], named entity recognition [13], [13], [14], and semantic role labeling [15]–[17]. Most of the research efforts in deep learning associated with NLP applications involve either *supervised learning*[1] or *unsupervised learning*[2].

This survey covers the emerging role of deep learning in the area of NLP, across a broad range of categories. The research presented in [18] is primarily focused on architectures, with little discussion of applications. On the other hand, this paper describes the challenges, opportunities, and evaluations of the impact of applying deep learning to NLP problems.

This survey has six sections, including this introduction. **Section 2** lays out the theoretical dimensions of NLP and artificial intelligence, and looks at deep learning as an approach to solving real-world problems. It motivates this study by addressing the question: Why use deep learning in NLP? The **third section** discusses fundamental concepts necessary to understand NLP, covering exemplary issues in representation, frameworks, and machine learning. The **fourth section** summarizes benchmark datasets employed in the NLP domain. **Section 5** focuses on some of the NLP applications where deep learning has demonstrated significant benefit. Finally, **Section 6** provides a conclusion, also addressing some open problems and promising areas for improvement.

II. Background

NLP has long been viewed as one aspect of artificial intelligence (AI), since understanding and generating natural language are high-level indications of intelligence. Deep learning is an effective AI tool, so we next situate deep learning in the AI world. After that we explain motivations for applying deep learning to NLP.

A. Artificial Intelligence and Deep Learning

There have been "islands of success" where big data are processed via AI capabilities to produce information to achieve critical operational goals (e.g., fraud detection). Accordingly, scientists and consumers anticipate enhancement across a

[1]Learning from training data to predict the type of new unseen test examples by mapping them to known pre-defined labels.

[2]Making sense of data without sticking to specific tasks and supervisory signals.

variety of applications. However, achieving this requires understanding of AI and its mechanisms and means (e.g., algorithms). Ted Greenwald, explaining AI to those who are not AI experts, comments: *"Generally AI is anything a computer can do that formerly was considered a job for a human"* [19].

An AI goal is to extend the capabilities of information technology (IT) from those to (1) generate, communicate, and store data, to also (2) process data into the knowledge that decision makers and others need [20]. One reason is that the available data volume is increasing so rapidly that it is now impossible for people to process all available data. This leaves two choices: (1) much or even most existing data must be ignored or (2) AI must be developed to process the vast volumes of available data into the essential pieces of information that decision-makers and others can comprehend. Deep learning is a bridge between the massive amounts of data and AI.

1) Definitions: Deep learning refers to *applying deep neural networks to massive amounts of data to learn a procedure aimed at handling a task*. The task can range from simple classification to complex reasoning. In other words, deep learning is *a set of mechanisms* ideally capable of deriving an optimum solution to any problem given a sufficiently extensive and relevant input dataset. Loosely speaking, deep learning is detecting and analyzing important structures/features in the data aimed at formulating a solution to a given problem. Here, AI and deep learning meet. One version of the goal or ambition behind AI is enabling a machine *to outperform what the human brain does*. Deep learning is a means to this end.

2) Deep Learning Architectures: Numerous deep learning architectures have been developed in different research areas, e.g., in NLP applications employing recurrent neural networks (RNNs) [21], convolutional neural networks (CNNs) [22], and more recently, recursive neural networks [23]. We focus our discussion on a review of the essential models, explained in relevant seminal publications.

Multi Layer Perceptron: A *multilayer perceptron* (MLP) has at least three layers (input, hidden, and output layers). A layer is simply a collection of neurons operating to transform information from the previous layer to the next layer. In the MLP architecture, the neurons in a layer do not communicate with each other. An MLP employs nonlinear activation functions. Every node in a layer connects to all nodes in the next layer, creating a fully connected network (Fig. 1). MLPs are the simplest type of *Feed-Forward Neural Networks* (FNNs). FNNs represent a general category of neural networks in which the connections between the nodes do not create any cycle, i.e., in a FNN there is no cycle of information flow.

Convolutional Neural Networks: Convolutional neural networks (CNNs), whose architecture is inspired by the human visual cortex, are a subclass of feed-forward neural networks. CNNs are named after the underlying mathematical operation, *convolution*, which yields a measure of the interoperability of its input functions. Convolutional neural networks are usually employed in situations where data is or needs to be represented with a 2D or 3D data map. In the data map representation, the proximity of data points usually corresponds to their information correlation.

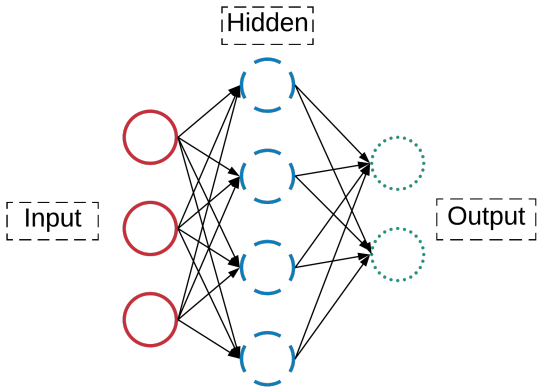

Fig. 1. The general architecture of a MLP.

In convolutional neural networks where the input is an image, the data map indicates that image pixels are highly correlated to their neighboring pixels. Consequently, the convolutional layers have 3 dimensions: width, height, and depth. That assumption possibly explains why the majority of research efforts dedicated to CNNs are conducted in the Computer Vision field [24].

A CNN takes an image represented as an array of numeric values. After performing specific mathematical operations, it represents the image in a new output space. This operation is also called feature extraction, and helps to capture and represent key image content. The extracted features can be used for further analysis, for different tasks. One example is image classification, which aims to categorize images according to some predefined classes. Other examples include determining which objects are present in an image and where they are located. See Fig. 2.

In the case of utilizing CNNs for NLP, the inputs are sentences or documents represented as matrices. Each row of the matrix is associated with a language element such as a word or a character. The majority of CNN architectures learn word or sentence representations in their training phase. A variety of CNN architectures were used in various classification tasks such as Sentiment Analysis and Topic Categorization [22], [25]–[27]. CNNs were employed for Relation Extraction and Relation Classification as well [28], [29].

Recurrent Neural Network: If we line up a sequence of FNNs and feed the output of each FNN as an input to the next one, a recurrent neural network (RNN) will be constructed. Like FNNs, layers in an RNN can be categorized into input, hidden, and output layers. In discrete time frames, sequences of input vectors are fed as the input, one vector at a time, e.g., after inputting each batch of vectors, conducting some operations and updating the network weights, the next input batch will be fed to the network. Thus, as shown in Fig. 3, at each time step we make predictions and use parameters of the current hidden layer as input to the next time step.

Hidden layers in recurrent neural networks can carry information from the past, in other words, memory. This characteristic makes them specifically useful for applications that deal with a sequence of inputs such as language modeling [30], i.e.,

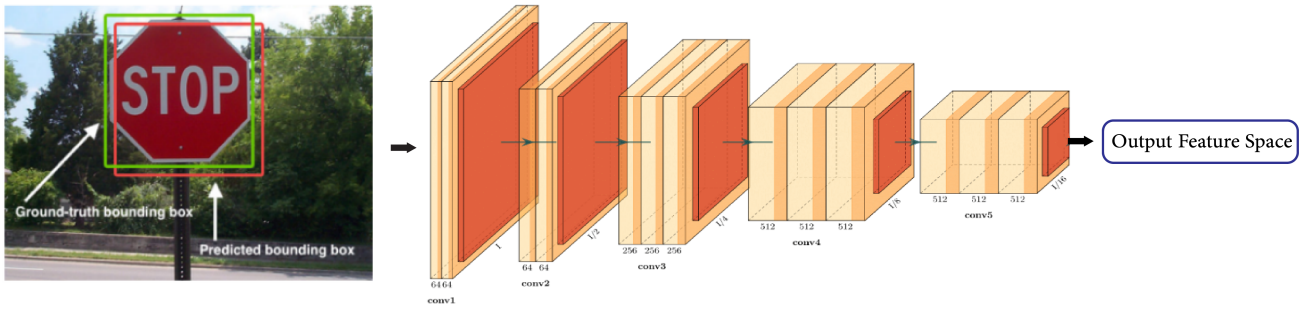

Fig. 2. A typical CNN architecture for object detection. The network provides a feature representation with attention to the specific region of an image (example shown on the left) that contains the object of interest. Out of the multiple regions represented (see an ordering of the image blocks, giving image pixel intensity, on the right) by the network, the one with the highest score will be selected as the main candidate.

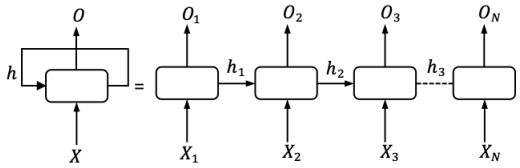

Fig. 3. Recurrent Neural Network (RNN), summarized on the left, expanded on the right, for N timesteps, with X indicating input, h hidden layer, and O output

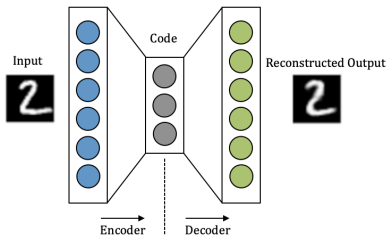

Fig. 4. Schematic of an Autoencoder

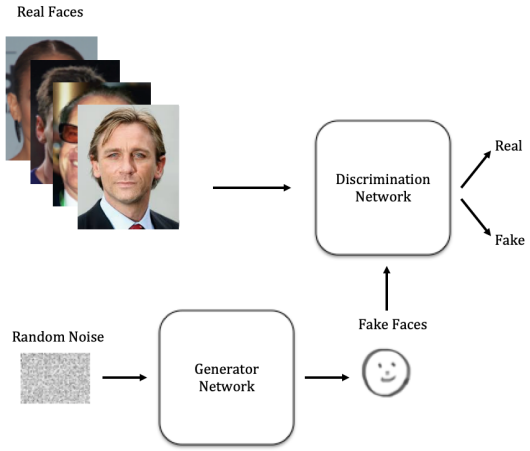

Fig. 5. Generative Adversarial Networks

representing language in a way that the machine understands. This concept will be described later in detail.

RNNs can carry rich information from the past. Consider the sentence: "Michael Jackson was a singer; some people consider him King of Pop." It's easy for a human to identify *him* as referring to Michael Jackson. The pronoun *him* happens seven words after *Michael Jackson;* capturing this dependency is one of the benefits of RNNs, where the hidden layers in an RNN act as memory units. Long Short Term Memory Network (LSTM) [31] is one of the most widely used classes of RNNs. LSTMs try to capture even long time dependencies between inputs from different time steps. Modern Machine Translation and Speech Recognition often rely on LSTMs.

Autoencoders: *Autoencoders* implement unsupervised methods in deep learning. They are widely used in dimensionality reduction[3] or NLP applications which consist of sequence to sequence modeling (see Section III-B [30]. Fig. 4 illustrates the schematic of an Autoencoder. Since autoencoders are unsupervised, there is no label corresponding to each input. They aim to learn a code representation for each input. The encoder is like a feed-forward neural network in which the input gets encoded into a vector (code). The decoder operates similarly to the encoder, but in reverse, i.e., constructing an output based on the encoded input. In data compression applications, we want the created output to be as close as possible to the original input. Autoencoders are *lossy*, meaning the output is an approximate reconstruction of the input.

Generative Adversarial Networks: Goodfellow [32] introduced *Generative Adversarial Networks (GANs)*. As shown in Fig. 5, a GAN is a combination of two neural networks, a discriminator and a generator. The whole network is trained in an iterative process. First, the generator network generates a fake sample. Then the discriminator network tries to determine whether this sample (ex.: an input image) is real or fake, i.e., whether it came from the real training data (data used for building the model) or not. The goal of the generator is to fool the discriminator in a way that the discriminator believes the artificial (i.e., generated) samples synthesized by the generator are real.

This iterative process continues until the generator produces samples that are indistinguishable by the discriminator. In

[3]Dimensionality reduction is an unsupervised learning approach which is the process of reducing the number of variables that were used to represent the data by identifying the most crucial information.

other words, the probability of classifying a sample as fake or real becomes like flipping a fair coin for the discriminator. The goal of the generative model is to capture the distribution of real data while the discriminator tries to identify the fake data. One of the interesting features of GANs (regarding being generative) is: once the training phase is finished, there is no need for the discrimination network, so we solely can work with the generation network. In other words, having access to the trained generative model is sufficient.

Different forms of GANs has been introduced, e.g., Sim GAN [7], Wasserstein GAN [33], info GAN [34], and DC GAN [35]. In one of the most elegant GAN implementations [36], entirely artificial, yet almost perfect, celebrity faces are generated; the pictures are not real, but fake photos produced by the network. In the NLP domain, GANs often are used for text generation [37], [38].

B. Motivation for Deep Learning in NLP

Deep learning applications are predicated on the choices of (1) feature representation and (2) deep learning algorithm alongside architecture. These are associated with data representation and learning structure, respectively. For data representation, surprisingly, there usually is a disjunction between what information is thought to be important for the task at hand, versus what representation actually yields good results. For instance, in sentiment analysis, lexicon semantics, syntactic structure, and context are assumed by some linguists to be of primary significance. Nevertheless, previous studies based on the bag-of-words (BoW) model demonstrated acceptable performance [39]. The bag-of-words model [40], often viewed as the vector space model, involves a representation which accounts only for the words and their frequency of occurrence. BoW ignores the order and interaction of words, and treats each word as a unique feature. BoW disregards syntactic structure, yet provides decent results for what some would consider syntax-dependent applications. This observation suggests that simple representations, when coupled with large amounts of data, may work as well or better than more complex representations. These findings corroborate the argument in favor of the importance of deep learning algorithms and architectures.

Often the progress of NLP is bound to effective language modeling. A goal of statistical language modeling is the probabilistic representation of word sequences in language, which is a complicated task due to the curse of dimensionality. The research presented in [41] was a breakthrough for language modeling with neural networks aimed at overcoming the curse of dimensionality by **(1)** learning a distributed representation of words and **(2)** providing a probability function for sequences.

A key challenge in NLP research, compared to other domains such as Computer Vision, seems to be the complexity of achieving an in-depth representation of language using statistical models. A primary task in NLP applications is to provide a representation of texts, such as documents. This involves feature learning, i.e., extracting meaningful information to enable further processing and analysis of the raw data.

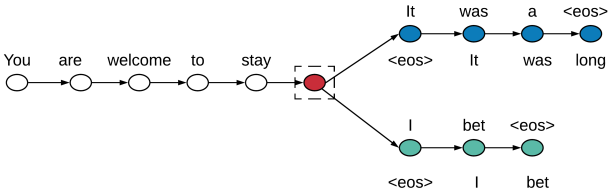

Fig. 6. Considering a given sequence, the skip-thought model generates the surrounding sequences using the trained encoder. The assumption is that the surrounding sentences are closely related, contextually.

Traditional methods begin with time-consuming hand-crafting of features, through careful human analysis of a specific application, and are followed by development of algorithms to extract and utilize instances of those features. On the other hand, deep supervised feature learning methods are highly data-driven and can be used in more general efforts aimed at providing a robust data representation.

Due to the vast amounts of unlabeled data, unsupervised feature learning is considered to be a crucial task in NLP. Unsupervised feature learning is, in essence, learning the features from unlabeled data to provide a low-dimensional representation of a high-dimensional data space. Several approaches such as K-means clustering and principal component analysis have been proposed and successfully implemented to this end. With the advent of deep learning and abundance of unlabeled data, unsupervised feature learning becomes a crucial task for representation learning, a precursor in NLP applications. Currently, most of the NLP tasks rely on annotated data, while a preponderance of unannotated data further motivates research in leveraging deep data-driven unsupervised methods.

Given the potential superiority of deep learning approaches in NLP applications, it seems crucial to perform a comprehensive analysis of various deep learning methods and architectures with particular attention to NLP applications.

III. Core Concepts in NLP

A. Feature Representation

Distributed representations are a series of compact, low dimensional representations of data, each representing some distinct informative property. For NLP systems, due to issues related to the atomic representation of the symbols, it is imperative to learn word representations.

At first, let's concentrate on how the features are represented, and then we focus on different approaches for learning word representations. The encoded input features can be characters, words [23], sentences [42], or other linguistic elements. Generally, it is more desirable to provide a compact representation of the words than a sparse one.

How to select the structure and level of text representation used to be an unresolved question. After proposing the word2vec approach [43], subsequently, doc2vec was proposed in [42] as an unsupervised algorithm and was called Paragraph Vector (PV). The goal behind PV is to learn fixed-length representations from variable-length text parts such as sentences and documents. One of the main objectives of doc2vec is

to overcome the drawbacks of models such as BoW and to provide promising results for applications such as text classification and sentiment analysis. A more recent approach is the skip-thought model which applies word2vec at the sentence-level [44]. By utilizing an encoder-decoder architecture, this model generates the surrounding sentences using the given sentence (Fig. 6). Next, let's investigate different kinds of feature representation.

1) One-Hot Representation: In one-hot encoding, each unique element that needs to be represented has its dimension which results in a very high dimensional, very sparse representation. Assume the words are represented with the one-hot encoding method. Regarding representation structure, there is no meaningful connection between different words in the feature space. For example, highly correlated words such as 'ocean' and 'water' will not be closer to each other (in the representation space) compared to less correlated pairs such as 'ocean' and 'fire.' Nevertheless, some research efforts present promising results using one-hot encoding [2].

2) Continuous Bag of Words: Continuous Bag-of-Words model (CBOW) has frequently been used in NLP applications. CBOW tries to predict a word given its surrounding context, which usually consists of a few nearby words [45]. CBOW is neither dependent on the sequential order of words nor necessarily on probabilistic characteristics. So it is not generally used for language modeling. This model is typically trained to be utilized as a pre-trained model for more sophisticated tasks. An alternative to CBOW is the weighted CBOW (WCBOW) [46] in which different vectors get different weights reflective of relative importance in context. The simplest example can be document categorization where features are words and weights are TF-IDF scores [47] of the associated words.

3) Word-Level Embedding: Word embedding is a learned representation for context elements in which, ideally, words with related semantics become highly correlated in the representation space. One of the main incentives behind word embedding representations is the high generalization power as opposed to sparse, higher dimensional representations [48]. Unlike the traditional bag-of-words model in which different words have entirely different representations regardless of their usage or collocations, learning a distributed representation takes advantage of word usage in context to provide similar representations for semantically correlated words. There are different approaches to create word embeddings. Several research efforts, including [43], [45], used random initialization by uniformly sampling random numbers with the objective of training an efficient representation of the model on a large dataset. This setup is intuitively acceptable for initialization of the embedding for common features such as part-of-speech tags. However, this may not be the optimum method for representation of less frequent features such as individual words. For the latter, pre-trained models, trained in a supervised or unsupervised manner, are usually leveraged for increasing the performance.

4) Character-Level Embedding: The methods mentioned earlier are mostly at higher levels of representation. Lower-level representations such as character-level representation require special attention as well, due to their simplicity of representation and the potential for correction of unusual character combinations such as misspellings [2]. For generating character-level embeddings, CNNs have successfully been utilized [10].

Character-level embeddings have been used in different NLP applications [49]. One of the main advantages is the ability to use small model sizes and represent words with lower-level language elements [10]. Here word embeddings are models utilizing CNNs over the characters. Another motivation for employing character-level embeddings is the out-of-vocabulary word (OOV) issue which is usually encountered when, for the given word, there is no equivalent vector in the word embedding. The character-level approach may significantly alleviate this problem. Nevertheless, this approach suffers from a weak correlation between characters and semantic and syntactic parts of the language. So, considering the aforementioned pros and cons of utilizing character-level embeddings, several research efforts tried to propose and implement higher-level approaches such as using sub-words [50] to create word embeddings for OOV instances as well as creating a semantic bridge between the correlated words [51].

B. Seq2Seq Framework

Most underlying frameworks in NLP applications rely on sequence-to-sequence (seq2seq) models in which not only the input but also the output is represented as a sequence. These models are common in various applications including machine translation[4], text summarization[5], speech-to-text, and text-to-speech applications[6].

The most common seq2seq framework is comprised of an encoder and a decoder. The encoder ingests the sequence of input data and generates a mid-level output which is subsequently consumed by the decoder to produce the series of final outputs. The encoder and decoder are usually implemented via a series of Recurrent Neural Networks or LSTM [31] cells.

The encoder takes a sequence of length T, $X = \{x_1, x_2, \cdots, x_T\}$, where $x_t \in V = \{1, \cdots, |V|\}$ is the representation of a single input coming from the vocabulary V, and then generates the output state h_t. Subsequently, the decoder takes the last state from the encoder, i.e., h_t, and starts generating an output of size L, $Y' = \{y'_1, y'_2, \cdots, y'_L\}$, based on its current state, s_t, and the ground-truth output y_t. In different applications, the decoder could take advantage of more information such as a context vector [52] or intra-attention vectors [53] to generate better outputs.

One of the most widely training approaches for seq2seq models is called *Teacher Forcing* [54]. Let us define $y = \{y_1, y_2, \cdots, y_L\}$ as the ground-truth output sequence correspondent to a given input sequence X. The model training

[4]The input is a sequence of words from one language (e.g., English) and the output is the translation to another language (e.g., French).

[5]The input is a complete document (sequence of words) and the output is a summary of it (sequence of words).

[6]The input is an audio recording of a speech (sequence of audible elements) and the output is the speech text (sequence of words).

based on the maximum-likelihood criterion employs the following cross-entropy (CE) loss minimization:

$$\mathcal{L}_{CE} = -\sum_{t=1}^{L} \log p_\theta(y_t|y_{t-1}, s_t, X) \tag{1}$$

where θ is the parameters of the model optimized during the training.

Once the model is optimized using the cross-entropy loss, it can generate an entire sequence as follows. Let $\hat{y}_t$ denote the output generated by the model at time t. Then, the next output is generated by:

$$\hat{y}_t = \arg\max_y p_\theta(y|\hat{y}_{t-1}, s_t) \tag{2}$$

In NLP applications, one can improve the output by using beam search to find a reasonably good output sequence [3]. During beam search, rather than using `argmax` for selecting the best output, we choose the top K outputs at each step, generate K different paths for the output sequence, and finally choose the one that provides better performance as the final output. Although, there has been some recent studies [55], [56] on improving the beam search by incorporating a similar mechanism during training of them model, studying this is outside the scope of this paper.

Given a series of the ground-truth output Y and the generated model output $\hat{Y}$, the model performance is evaluated using a task-specific measures such as ROUGE [57], BLEU [58], and METEOR [59]. As an example, ROUGE$_L$, which is an evaluation metric in NLP tasks, uses the largest common substring between ground-truth Y and model output $\hat{Y}$ to evaluate the generated output.

C. Reinforcement Learning in NLP

Although the seq2seq models explained in Section III-B achieve great successes w.r.t. traditional methods, there are some issues with how these models are trained. Generally speaking, seq2seq models like the ones used in NLP applications face two issues: (1) *exposure bias* and (2) *inconsistency between training time and test time measurements* [60].

Most of the popular seq2seq models are minimizing cross-entropy loss as their optimization objective via Teacher Forcing (Section III-B). In teacher forcing, during the training of the model, the decoder utilizes two inputs, the former decoder output state s_{t-1} and the ground-truth input y_t, to determine its current output state s_t. Moreover, it employs them to create the next token, i.e., $\hat{y}_t$. However, at test time, the decoder fully relies on the previously created token from the model distribution. As the ground-truth data is not available, such a step is necessary to predict the next action. Henceforth, in training, the decoder input is coming from the ground truth, while, in the test phase, it relies on the previous prediction.

This *exposure bias* [61] induces error growth through output creation at the test phase. One approach to remedy this problem is to remove the ground-truth dependency in training by solely relying on model distribution to minimize the cross-entropy loss. Scheduled sampling [54] is one popular method to handle this setback. During scheduled sampling, we first pre-train the model using cross-entropy loss and then slowly replace the ground-truth with samples the model generates.

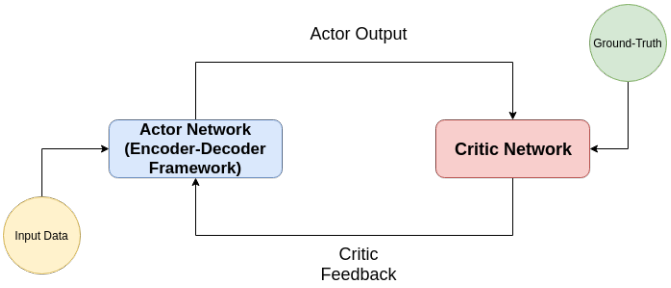

Fig. 7. A simple Actor-Critic framework.

The second obstacle with seq2seq models is that, when training is finished using the cross-entropy loss, it is typically evaluated using non-differentiable measures such as ROUGE or METEOR. This will form an inconsistency between the training objective and the test evaluation metric. Recently, it has been demonstrated that both of these problems can be tackled by utilizing techniques from reinforcement learning [60].

Among most of the well-known models in reinforcement learning, policy gradient techniques [62] such as the REINFORCE algorithm [63] and actor-critic based models such as value-based iteration [64], and Q-learning [65], are among the most common techniques used in deep learning in NLP.

Using the model predictions (versus the ground-truth) for the sequence to sequence modeling and generation, at training time, was initially introduced by Daume *et al.* [66]. According to their approach, SEARN, the structured prediction can be characterized as one of the reinforcement learning cases as follows: *The model employs its predictions to produce a sequence of actions (words sequences). Then, at each time step, a greedy search algorithm is employed to learn the optimal action, and the policy will be trained to predict that particular action*.

In Actor-Critic training, the actor is usually the same neural network used to generate the output, while the critic is a regression model that estimates how the actor performed on the input data. The actor later receives the feedback from the critic and improves its actions. Fig 7 shows this framework. It is worth noting that action in most of the NLP-related applications is like selecting the next output token while the state is the decoder output state at each stage of decoding. These models have mostly been used for robotic [67] and Atari games [68] due to the small action space in these applications. However, when we use them in NLP applications, they face multiple challenges. The action space in most of the NLP applications could be defined as the number of tokens in the vocabulary (usually between 50K to 150K tokens). Comparing this to the action space in a simple Atari game, which on average has less than 20 actions [68], shows why these Actor-Critic models face difficulties when applied to NLP applications. A major challenge is the massive action space in NLP applications, which not only causes difficulty for the right action selection, but also will make the training process very slow. This makes the process of finding the best

Actor-Critic model very complicated and model convergence usually requires a lot of tweaks to the models.

IV. Datasets

Many different researchers for different tasks use benchmark datasets, such as those discussed below. Benchmarking in machine learning refers to the assessment of methods and algorithms, comparing those regarding their capability to learn specific patterns. Benchmarking aids validation of a new approach or practice, relative to other existing methods.

Benchmark datasets typically take one of three forms.

1) The first is real-world data, obtained from various real-world experiments.
2) The second is synthetic data, artificially generated to mimic real-world patterns. Synthetic data is generated for use instead of real data. Such datasets are of special interest in applications where the amount of data required is much larger than that which is available, or where privacy considerations are crucial and strict, such as in the healthcare domain.
3) The third type are toy datasets, used for demonstration and visualization purposes. Typically they are artificially generated; often there is no need to represent real-world data patterns.

The foundation of Deep Learning utilization is the availability of data to teach the system about pattern identification. The effectiveness of the model depends on the quality of the data. Despite the successful implementation of universal language modeling techniques such as BERT [69], however, such models can be used solely for pre-training the models. Afterward, the model needs to be trained on the data associated with the desired task. Henceforth, based on the everyday demands in different machine domains such as NLP, creating new datasets is crucial.

On the other hand, creating new datasets is not usually an easy matter. Informally speaking, the newly created dataset should be: the right data to train on, sufficient for the evaluation, and accurate to work on. Answering the questions of "what is the meaning of right and accurate data" is highly application-based. Basically, the data should have sufficient information, which depends on the quality and quantity of the data.

To create a dataset, the first step is always asking "what are we trying to do and what problem do we need to solve?" and "what kind of data do we need and how much of it is required?" The next step is to create training and testing portions. The training data set is used to train a model to know how to find the connections between the inputs and the associated outputs. The test data set is used to assess the intelligence of the machine, i.e., how well the trained model can operate on the unseen test samples. Next, we must conduct data preparation to make sure the data and its format is simple and understandable for human experts. After that, the issue of data accessibility and ownership may arise. Distribution of data may need to have specific authorizations, especially if we are dealing with sensitive or private data.

Given the aforementioned roadmap, creating proper datasets is complicated and of great importance. That's why few datasets are frequently chosen by the researchers and developers for benchmarking. A summary of widely used benchmark datasets is provided in Table I.

V. Deep Learning for NLP Tasks

This section describes NLP applications using deep learning. Fig. 8 shows representative NLP tasks (and the categories they belong to). A fundamental question is: *"How can we evaluate an NLP algorithm, model, or system?"* In [70], some of the most common evaluation metrics have been described. This reference explains the fundamental principles of evaluating NLP systems.

A. Basic Tasks

1) Part-Of-Speech Tagging: Part-of-Speech tagging is one of the basic tasks in Natural Language Processing. It is the process of labeling words with their part of speech categories. Part of speech is leveraged for many crucial tasks such as named entity recognition. One commonly used dataset for Part-of-Speech tagging is the WSJ corpus[7]. This dataset contains over a million tokens and has been utilized widely as a benchmark dataset for the performance assessment of POS tagging systems. Traditional methods are still performing very well for this task [12]. However, neural network based methods have been proposed for Part-of-Speech tagging [71].

For example, the deep neural network architecture named *CharWNN* has been developed to join word-level and character-level representations using convolutional neural networks for POS tagging [10]. The emphasis in [10] is the importance of character-level feature extraction as their experimental results show the necessity of employing hand-crafted features in the absence of character-level features for achieving the state-of-the-art. In [72], a wide variety of neural network based models have been proposed for sequence tagging tasks, e.g., LSTM networks, bidirectional LSTM networks, LSTM networks with a CRF[8] layer, etc. Sequence tagging itself includes part of speech tagging, chunking, and named entity recognition. Likewise, a globally normalized transition-based neural network architecture has been proposed for POS-tagging [73]. State-of-the-art results are summarized in Table II.

2) Parsing: Parsing is assigning a structure to a recognized string. There are different types of parsing. *Constituency Parsing* refers in particular to assigning a syntactic structure to a sentence. A greedy parser has been introduced in [81] which performs a syntactic and semantic summary of content using vector representations. To enhance the results achieved by [81], the approach proposed in [82] focuses on learning morphological embeddings. Recently, deep neural network models outperformed traditional algorithms. State-of-the-art results are summarized in Table III.

Another type of parsing is called *Dependency Parsing*. Dependency structure shows the structural relationships between the words in a targeted sentence. In dependency parsing,

[7]Penn Treebank Wall Street Journal (WSJ-PTB).
[8]Conditional Random Field.

TABLE I
BENCHMARK DATASETS.

Task	Dataset	Link
Machine Translation	WMT 2014 EN-DE WMT 2014 EN-FR	http://www-lium.univ-lemans.fr/~schwenk/cslm_joint_paper/
Text Summarization	CNN/DM Newsroom DUC Gigaword	https://cs.nyu.edu/~kcho/DMQA/ https://summari.es/ https://www-nlpir.nist.gov/projects/duc/data.html https://catalog.ldc.upenn.edu/LDC2012T21
Reading Comprehension Question Answering Question Generation	ARC CliCR CNN/DM NewsQA RACE SQuAD Story Cloze Test NarativeQA Quasar SearchQA	http://data.allenai.org/arc/ http://aclweb.org/anthology/N18-1140 https://cs.nyu.edu/~kcho/DMQA/ https://datasets.maluuba.com/NewsQA http://www.qizhexie.com/data/RACE_leaderboard https://rajpurkar.github.io/SQuAD-explorer/ http://aclweb.org/anthology/W17-0906.pdf https://github.com/deepmind/narrativeqa https://github.com/bdhingra/quasar https://github.com/nyu-dl/SearchQA
Semantic Parsing	AMR parsing ATIS (SQL Parsing) WikiSQL (SQL Parsing)	https://amr.isi.edu/index.html https://github.com/jkkummerfeld/text2sql-data/tree/master/data https://github.com/salesforce/WikiSQL
Sentiment Analysis	IMDB Reviews SST Yelp Reviews Subjectivity Dataset	http://ai.stanford.edu/~amaas/data/sentiment/ https://nlp.stanford.edu/sentiment/index.html https://www.yelp.com/dataset/challenge http://www.cs.cornell.edu/people/pabo/movie-review-data/
Text Classification	AG News DBpedia TREC 20 NewsGroup	http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html https://wiki.dbpedia.org/Datasets https://trec.nist.gov/data.html http://qwone.com/~jason/20Newsgroups/
Natural Language Inference	SNLI Corpus MultiNLI SciTail	https://nlp.stanford.edu/projects/snli/ https://www.nyu.edu/projects/bowman/multinli/ http://data.allenai.org/scitail/
Semantic Role Labeling	Proposition Bank OneNotes	http://propbank.github.io/ https://catalog.ldc.upenn.edu/LDC2013T19

TABLE II
POS TAGGING STATE-OF-THE-ART MODELS EVALUATED ON THE WSJ-PTB DATASET.

Model	Accuracy
Character-aware neural language models [74]	97.53
Transfer Learning + GRU [75]	97.55
Bi-directional LSTM + CNNs + CRF [76]	97.55
Adversarial Training + Bi-LSTM [77]	97.59
Character Composition + Bi-LSTM [78]	97.78
String Embedding + LSTM [79]	97.85
Meta-BiLSTM [80]	**97.96**

TABLE III
CONSTITUENCY PARSING STATE-OF-THE-ART MODELS EVALUATED ON THE WSJ-PTB DATASET.

Model	Accuracy
Recurrent neural network grammars (RNNG) [83]	93.6
In-order traversal over syntactic trees + LSTM [84]	94.2
Model Combination and Reranking [85]	94.6
Self-Attentive Encoder [86]	**95.1**

phrasal elements and phrase-structure rules do not contribute to the process. Rather, the syntactic structure of the sentence is expressed only in terms of the words in the sentence and the associated relations between the words.

Neural networks have shown their superiority regarding generalizability and reducing the feature computation cost. In [87], a novel neural network-based approach was proposed for a transition-based dependency parser. Neural network based models that operate on task-specific transition systems have also been utilized for dependency parsing [73]. A regularized parser with bi-affine classifiers has been proposed for the prediction of arcs and labels [88]. Bidirectional-LSTMs have been used in dependency parsers for feature representation [89]. A new control structure has been introduced for sequence-to-sequence neural networks based on the stack LSTM and has been used in transition-based parsing [90].

3) Semantic Role Labeling: Semantic Role Labeling (SRL) is the process of identification and classification of text arguments. It is aimed at the characterization of elements to determine "who" did "what" to "whom" as well as "how," "where," and "when." It identifies the predicate-argument structure of a sentence. The predicate, in essence, refers to "what," while the arguments consist of the associated participants and properties in the text. The goal of SRL is to extract the semantic relations between the predicate and the related arguments.

Most of the previously-reported research efforts are based

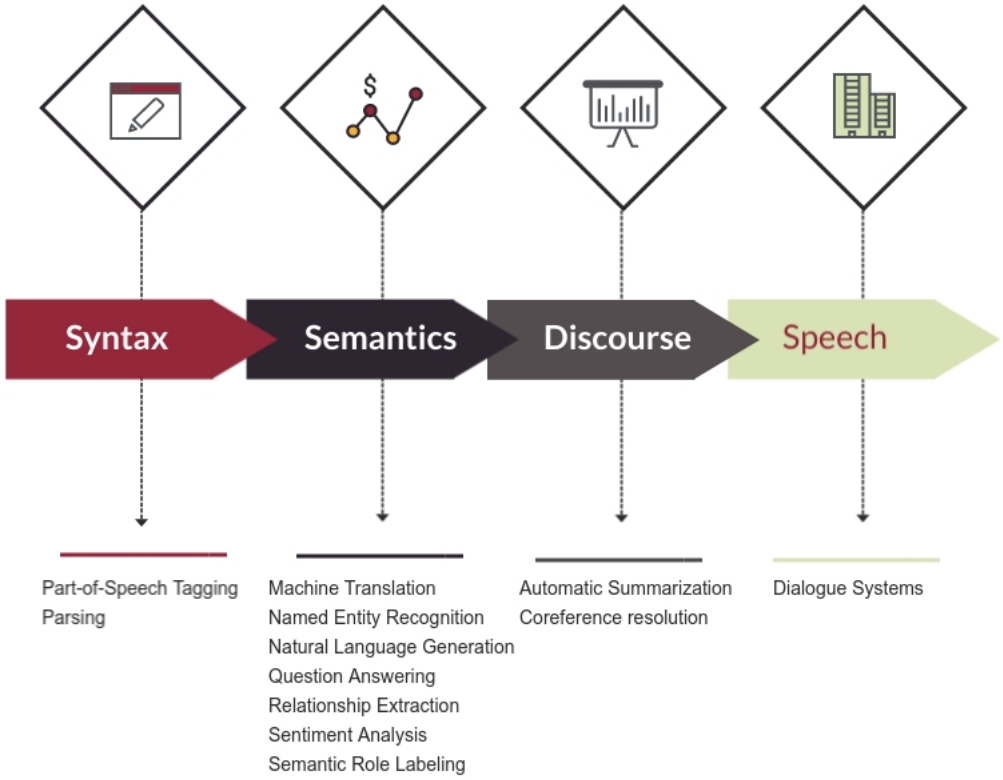

Fig. 8. NLP tasks investigated in this study.

on explicit representations of semantic roles. Recently, deep learning approaches have achieved the SRL state-of-the-art without taking the explicit syntax representation into consideration [91]. On the other hand, it is argued that the utilization of syntactic information can be leveraged to improve the performance of syntactic-agnostic[9] models [92]. A linguistically-informed self-attention (LISA) model has been proposed to leverage both multi-task learning and self-attention for effective utilization of the syntactic information for SRL [93]. Current state-of-the-art methods employ joint prediction of predicates and arguments [94], novel word representation approaches [95], and self-attention models [96]; see Table IV.

TABLE IV
SEMANTIC ROLE LABELING CURRENT STATE-OF-THE-ART MODELS EVALUATED ON THE ONTONOTES DATASET [97]. THE ACCURACY METRIC IS F_1 SCORE.

Model	Accuracy (F_1)
Self-Attention + RNN [96]	83.9
Contextualized Word Representations [95]	84.6
Argumented Representations + BiLSTM [94]	**85.3**

[9]Note that being syntactic-agnostic does not imply discarding syntactic information. It means they are not explicitly employed.

B. Text Classification

The primary objective of text classification is to assign predefined categories to text parts (which could be a word, sentence, or whole document) for preliminary classification purposes and further organization and analysis. A simple example is the categorization of given documents as to political or non-political news articles.

The use of CNNs for sentence classification, in which training the model on top of pretrained word-vectors through fine-tuning, has resulted in considerable improvements in learning task-specific vectors [22]. Later, a Dynamic Convolutional Neural Network (DCNN) architecture – essentially a CNN with a dynamic k-max pooling method – was applied to capture the semantic modeling of sentences [98]. In addition to CNNs, RNNs have been used for text classification. An LSTM-RNN architecture has been utilized in [99] for sentence embedding with particular superiority in a defined web search task. A Hierarchical Attention Network (HAN) has been utilized to capture the hierarchical structure of text, with a word-level and sentence-level attention mechanism [100].

Some models used the combination of both RNNs and CNNs for text classification such as [101]. This is a recurrent architecture in addition to max-pooling with an effective word representation method, and demonstrates superiority compared to simple window-based neural network approaches. Another unified architecture is the C-LSTM proposed in [102] for

sentence and document modeling in classification. Current state-of-the-art methods are summarized in Table V.

TABLE V
THE CLASSIFICATION ACCURACY OF STATE-OF-THE-ART METHODS, EVALUATED ON THE AG NEWS CORPUS DATASET [2].

Model	Accuracy
CNN [103]	91.33
Deep Pyramid CNN [104]	93.13
CNN [105]	93.43
Universal Language Model Fine-tuning (ULMFiT) [106]	**94.99**

C. Information Extraction

Information extraction identifies structured information from "unstructured" data such as social media posts and online news. Deep learning has been utilized for information extraction regarding subtasks such as *Named Entity Recognition*, *Relation Extraction*, *Coreference Resolution*, and *Event Extraction*.

1) Named Entity Recognition: Named Entity Recognition (NER) aims to locate and categorize named entities in context into pre-defined categories such as the names of people and places. The application of deep neural networks in NER has been investigated by the employment of CNN [107] and RNN architectures [108], as well as hybrid bidirectional LSTM and CNN architectures [14]. NeuroNER [109], a named-entity recognition tool, operates based on artificial neural networks. State-of-the-art models are reported in Table VI.

TABLE VI
STATE OF THE ART MODELS REGARDING NAME ENTITY RECOGNITION. EVALUATION IS PERFORMED ON THE CONLL-2003 SHARED TASK DATASET [110]. THE EVALUATION METRIC IS F_1 SCORE.

Model	Accuracy
Semi-supervised Sequence Modeling [111]	92.61
Google BERT [112]	92.8
Contextual String Embeddings [79]	**93.09**

2) Relation Extraction: Relation Extraction aims to find the semantic relationships between entity pairs. The recursive neural network (RNN) model has been proposed for semantic relationship classification by learning compositional vector representations [113]. For relation classification, CNN architectures have been employed as well, by extracting lexical and sentence level features [28].

3) Coreference Resolution: Coreference resolution includes identification of the mentions in a context that refer to the same entity. For instance, the mentions "car," "Camry," and "it" could all refer to the same entity. For the first time in [114], Reinforcement Learning (RL) was applied to coreference resolution. Current state-of-the-art methods leverage an attention mechanism [115].

4) Event Extraction: A specific type of extracted information from text is an event. Such extraction may involve recognizing trigger words related to an event and assigning labels to entity mentions that represent event triggers. Convolutional neural networks have been utilized for event detection; they handle problems with feature-based approaches including exhaustive feature engineering and error propagation phenomena for feature generation [116]. In 2018, Nguyen and Grishman applied graph-CNN (GCCN) where the convolutional operations are applied to syntactically dependent words as well as consecutive words [117]; their adding entity information reflected the state-of-the-art using CNN models.

D. Sentiment analysis

The primary goal in sentiment analysis is the extraction of subjective information from text by contextual mining. Sentiment analysis is considered high-level reasoning based on source data. Sentiment analysis is sometimes called opinion mining, as its primary goal is to analyze human opinion, sentiments, and even emotions regarding products, problems, and varied subjects. Important works on sentiment analysis or opinion mining include [118], [119]. Since 2000, much attention has been given to sentiment analysis, due to its relation to a wide variety of applications, its associations with new research challenges, and the availability of abundant data.

A critical aspect of research in sentiment analysis is content granularity. Considering this criterion, sentiment analysis is generally divided into three categories/levels: document level, sentence level, and aspect level.

1) Document-level Sentiment Analysis: At the document level, the task is to determine whether the whole document reflects a positive or negative sentiment about exactly one entity. This differs from opinion mining regarding multiple entries. The Gated Recurrent Neural Network architecture has been utilized successfully for effectively encoding the sentences' relations in the semantic structure of the document [120]. Domain adaptation has been investigated as well, to deploy the trained model on unseen new sources [121].

2) Sentence-level Sentiment Analysis: At the sentence-level, sentiment analysis determines the positivity, negativity, or neutrality regarding an opinion expressed in a sentence. One general assumption for sentence-level sentiment classification is the existence of only one opinion from a single opinion holder in an expressed sentence. Recursive autoencoders have been employed for sentence-level sentiment label prediction by learning the vector space representations for phrases [122]. Long Short-Term Memory (LSTM) recurrent models have also been utilized for tweet sentiment prediction [123]. The Sentiment Treebank and Recursive Neural Tensor Networks [124] have shown promise for predicting fine-grained sentiment labels.

3) Aspect-level Sentiment Analysis: Document-level and sentence-level sentiment analysis usually focus on the sentiment itself, not the target of the sentiment, e.g., a product. Aspect-level sentiment analysis directly targets an opinion, with the assumption of the existence of the sentiment and its target. A document or sentence may not have a generally positive or negative sentiment, but may have multiple subparts with different targets, each with a positive or negative sentiment. This can make aspect-level analysis even more challenging than other types of sentiment categorization.

Aspect-level sentiment analysis usually involves *Aspect Sentiment Classification* and *Aspect Extraction*. The former determines opinions on different aspects (positive, neutral, or negative) while the latter identifies the target aspect for evaluation in context. As an example consider the following sentence: *"This car is old. It must be repaired and sold!"*. "This car" is what is subject to evaluation and must be extracted first. Here, the opinion about this aspect is negative.

For aspect-level sentiment classification, attention-based LSTMs are proposed to connect the aspect and sentence content for sentiment classification [125]. For aspect extraction, deep learning has successfully been proposed in opinion mining [126]. State-of-the-art methods rely on converting aspect-based sentiment analysis to sentence-pair classification tasks [69], post-training approaches [127] on the popular language model BERT [112], and employment of pre-trained embeddings [128].

E. Machine Translation

Machine Translation (MT) is one of the areas of NLP that has been profoundly affected by the advances in deep learning. The first subsection below explains methods used in the pre-deep learning period, as explained in reference NLP textbooks such as "Speech and Language Processing" [129]. The remainder of this section is dedicated to delving into recent innovations in MT which are based on neural networks, started by [130].

1) Traditional Machine Translation: One of the first demonstrations of machine translation happened in 1954 [131] in which the authors tried to translate from Russian to English. This translation system was based on six simple rules, but had a very limited vocabulary. It was not until the 1990s that successful statistical implementations of machine translation emerged as more bilingual corpora became available [129]. In [58] the BLEU score was introduced as a new evaluation metric, allowing more rapid improvement than when the only approach involved using human labor for evaluation.

2) Neural Machine Translation: It was after the success of the neural network in image classification tasks that researchers started to use neural networks in machine translation (NMT). Around 2013, research groups started to achieve breakthrough results in NMT. Unlike traditional statistical machine translation, NMT is based on an end-to-end neural network [132]. This implies that there is no need for extensive preprocessing and word alignments. Instead, the focus shifted toward network structure.

Fig. 11 shows an example of an end-to-end recurrent neural network for machine translation. A sequence of input tokens is fed into the network. Once it reaches an end-of-sentence (EOS) token, it starts generating the output sequence. The output sequence is generated in the same recurrent manner as the input sequence until it reaches an end-of-sentence token. One major advantage of this approach is that there is no need to specify the length of the sequence; the network takes it into account automatically. In other words, the end-of-sentence token determines the length of the sequence. Networks implicitly learn that longer input sentences usually lead to longer

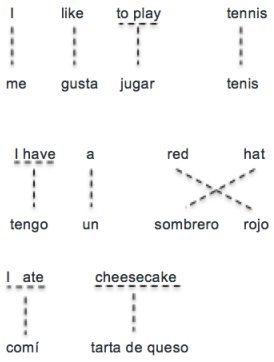

Fig. 9. Alignment in Machine Translation

output sentences with varying length, and that ordering can change. For instance, the second example in Fig. 9 shows that adjectives generally come before nouns in English but after nouns in Spanish. There is no need to explicitly specify this since the network can capture such properties. Moreover, the amount of memory that is used by NMT is just a fraction of the memory that is used in traditional statistical machine translation [133].

[130] was one of the early works that incorporated recurrent neural networks for machine translation. They were able to achieve a perplexity (a measure where lower values indicate better models) that was 43% less than the state-of-the-art alignment based translation models. Their recurrent continuous translation model (RCTM) is able to capture word ordering, syntax, and meaning of the source sentence explicitly. It maps a source sentence into a probability distribution over sentences in the target language. RCTM estimates the probability $P(f|e)$ of translating a sentence $e = e_1 + ... + e_k$ in the source language to target language sentence $f = f_1 + ... + f_m$. RCTM estimates $P(f|e)$ by considering source sentence e as well as the preceding words in the target language $f_{1:i-1}$:

$$P(f|e) = \prod_{i=1}^{m} P(f_i|f_{1:i-1}, e) \tag{3}$$

The representation generated by RCTM acts on n-grams in the lower layers, and acts more on the whole sentence as one moves to the upper layers. This hierarchical representation is performed by applying different layers of convolution. First a continuous representation of each word is generated; i.e., if the sentence is $e = e_1...e_k$, the representation of the word e_i will be $v(e_i) \in \mathbb{R}^{q\times 1}$. This will result in sentence matrix $\mathbf{E}^e \in \mathbb{R}^{q\times k}$ in which $\mathbf{E}^e_{:,i} = v(e_i)$. This matrix representation of the sentence will be fed into a series of convolution layers in order to generate the final representation $\mathbf{e}$ for the recurrent neural network. The approach is illustrated in Fig. 10. Equations for the pipeline are as follows.

$$s = \mathbf{S}.csm(e) \tag{4}$$

$$h_1 = \sigma(\mathbf{I}.v(f_1) + \mathbf{s}) \tag{5}$$

$$h_{i+1} = \sigma(\mathbf{R}.h_i + \mathbf{I}.v(f_{i+1}) + \mathbf{s}) \tag{6}$$

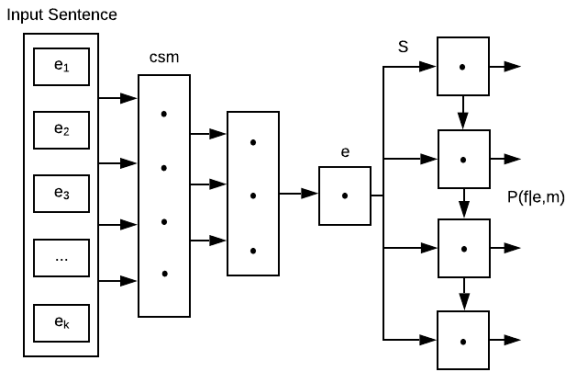

Fig. 10. Recurrent Continuous Translation Models (RCTM) [130].

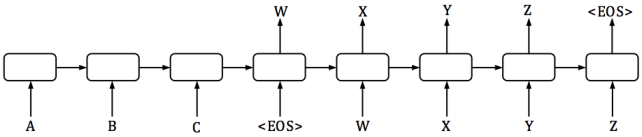

Fig. 11. Sequence to sequence learning with LSTM.

$$o_{i+1} = \mathbf{O}.h_i \tag{7}$$

In order to take into account the sentence length, the authors introduced RCTM II which estimates the length of the target sentence. RCTM II was able to achieve better perplexity on WMT datasets (see top portion of Table I) than other existing machine translation systems.

In another line of work, [134] presented an end-to-end sequence learning approach without heavy assumptions on the structure of the sequence. Their approach consists of two LSTMs, one for mapping the input to a vector of fixed dimension and another LSTM for decoding the output sequence from the vector. Their model was able to handle long sentences as well as sentence representations that are sensitive to word order. As shown in Fig. 11, the model reads "ABC" as an input sequence and produces "WXYZ" as output sequence. The $< EOS >$ token indicates the end of prediction. The network was trained by maximizing the log probability of the translation (η) given the input sequence (ζ). In other words, the objective function is:

$$1/|\mathcal{D}| \sum_{(\eta,\zeta)\in\mathcal{D}} logP(\eta|\zeta) \tag{8}$$

$\mathcal{D}$ is the training set and $|\mathcal{D}|$ is its size. One of the novelties of their approach was reversing word order of the source sentence. This helps the LSTM to learn long term dependencies. Having a fixed-length vector in the decoder phase is one of the bottlenecks of the encoder-decoder approach. [132] argues that a network will have a hard time compressing all the information from the input sentence into a fixed-size vector. They address this by allowing the network to search

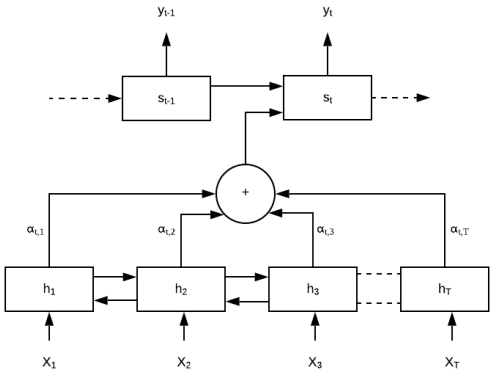

Fig. 12. Attention Mechasim for Neural Machine Translation [132].

segments of the source sentence that are useful for predicting the translation. Instead of representing the input sentence as a fixed-size vector, in [132] the input sentence is encoded to a sequence of vectors and a subset of them is chosen by using a method called attention mechanism as shown in Fig. 12.

In their approach $P(y_i|y_1, ..., y_{i-1}, X) = g(y_{i-1}, s_i, c_i)$, in which $s_i = f(s_{i-1}, y_{i-1}, c_i)$. While previously c was the same for all time steps, here c takes a different value, c_i, at each time step. This accounts for the attention mechasim (context vector) around that specific time step. c_i is computed according to the following:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j, \;\; \alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T_x} exp(e_{ik})}, \;\; e_{ij} = a(s_{i-1}, h_j)$$

.

Here a is the alignment model that is represented by a feed forward neural network. Also $h_j = [\overrightarrow{h_j^T}, \overleftarrow{h_j^T}]$, which is a way to include information both about preceding and following words in h_j. The model was able to outperform the simple encoder-decoder approach regardless of input sentence length.

Improved machine translation models continue to emerge, driven in part by the growth in people's interest and need to understand other languages Most of them are variants of the end-to-end decoder-encoder approach. For example, [135] tries to deal with the problem of rare words. Their LSTM network consists of encoder and decoder layers using residual layers along with the attention mechanism. Their system was able to decrease training time, speed up inference, and handle translation of rare words. Comparisons between some of the state-of-the-art neural machine translation models are summarized in Table VII.

F. Question Answering

Question answering (QA) is a fine-grained version of Information Retrieval (IR). In IR a desired set of information has to be retrieved from a set of documents. The desired information could be a specific document, text, image, etc. On the other hand, in QA specific answers are sought, typically ones that

TABLE VII
THE MACHINE TRANSLATION STATE-OF-THE-ART MODELS EVALUATED ON THE *English-German dataset of ACL 2014 Ninth Workshop on Statistical Machine TRranslation*. THE EVALUATION METRIC IS $BLEU$ SCORE.

Model	Accuracy
Convolutional Seq-to-Seq [136]	25.2
Attention Is All You Need [137]	28.4
Weighted Transformer [138]	28.9
Self Attention [139]	29.2
DeepL Translation Machine [10]	33.3
Back-translation [140]	**35.0**

can be inferred from available documents. Other areas of NLP such as reading comprehension and dialogue systems intersect with question answering.

Research in computerized question answering has proceeded since the 1960s. In this section, we present a general overview of question answering system history, and focus on the breakthroughs in the field. Like all other fields in NLP, question answering was also impacted by the advancement of deep learning [141], so we provide an overview of QA in deep learning contexts. We briefly visit visual question answering as well.

1) Rule-based Question Answering: Baseball [142] is one of the early works (1961) on QA where an effort was made to answer questions related to baseball games by using a game database. The baseball system consists of (1) question read-in, (2) dictionary lookup for words in the question, (3) syntactic (POS) analysis of the words in question, (4) content analysis for extracting the input question, and (5) estimating relevance regarding answering the input question.

IBM's [143] statistical question answering system consisted of four major components:

1) Question/Answer Type Classification
2) Query Expansion/Information Retrieval
3) Name Entity Making
4) Answer Selection

Some QA systems fail when semantically equivalent relationships are phrased differently. [144] addressed this by proposing fuzzy relation matching based on mutual information and expectation maximization.

2) Question answering in the era of deep learning: Smartphones (Siri, Ok Google, Alexa, etc.) and virtual personal assistants are common examples of QA systems with which many interact on a daily basis. While earlier such systems employed rule-based methods, today their core algorithm is based on deep learning. Table VIII presents some questions and answers provided by Siri on an iPhone.

TABLE VIII
TYPICAL QUESTION ANSWERING PERFORMANCE BASED ON DEEP LEARNING.

Question	**Answer**
Who invented polio vaccine?	*The answer I found is Jonas Salk*
Who wrote Harry Potter?	*J.K.Rowling wrote Harry Potter in 1997*
When was Einstein born?	*Albert Einstein was born March 14, 1879*

[146] was one of the first machine learning based papers that reported results on QA for a reading comprehension test. The system tries to pick a sentence in the database that has an answer to a question, and a feature vector represents each question-sentence pair. The main contribution of [146] is proposing a feature vector representation framework which is aimed to provide information for learning the model. There are five classifiers (location, date, etc.), one for each type of question. They were able to achieve accuracy competitive with previous approaches.

As illustrated in Fig. 13, [145] uses convolutional neural networks in order to encode Question-Answer sentence pairs in the form of fixed length vectors regardless of the length of the input sentence. Instead of using distance measures like cosine correlation, they incorporate a non-linear tensor layer to match the relevance between question and answer. Equation 9 calculates the matching degree between question q and its corresponding answer a.

$$s(q,a) = \mathbf{u}^T \mathbf{f}(\mathbf{v}_q^T \mathbf{M}^{[1:r]} \mathbf{v}_a + \mathbf{V} \begin{bmatrix} \mathbf{v}_q \\ \mathbf{v}_a \end{bmatrix} + \mathbf{b}) \tag{9}$$

$\mathbf{f}$ is the standard element-wise non-linearity function, $\mathbf{M}^{[1:r] \in R^{n_s \times n_s \times r}}$ is a tensor, $\mathbf{V} \in R^{r \times 2n_s}$, $\mathbf{b} \in R^r$, $\mathbf{u} \in R^r$.

The model tries to capture the interaction between question and answer. Inspired by findings in neuroscience, [71] incorporated episodic memory[11] in their Dynamic Memory Network (DMN). By processing input sequences and questions, DMN forms episodic memories to answer relevant questions. As illustrated in Fig. 14, their system is trained based on raw Input-Question-Answer triplets.

DMN consists of four modules that communicate with each other as shown in Fig. 15. The **input module** encodes raw input text into a distributed vector representation; likewise the **question module** encodes a question into its distributed vector representation. The **episodic memory module** uses the attention mechanism in order to focus on a specific part of the input module. Through an iterative process, this module produces a *memory* vector representation that considers the question as well as previous memory. The **answer module** uses the final memory vector to generate an answer. The model improved upon state-of-the-art results on tasks such as the ones shown in Fig. 14. DMN is one of the architectures that could potentially be used for a variety of NLP applications such as classification, question answering, and sequence modeling.

[147] introduced a Dynamic Coattention Network (DCN) in order to address local maxima corresponding to incorrect answers; it is considered to be one of the best approaches to question answering.

3) Visual Question Answering: Given an input image, Visual Question Answering (VQA) tries to answer a natural language question about the image [148]. VQN addresses multiple problems such as object detection, image segmentation, sentiment analysis, etc. [148] introduced the task of VQA by providing a dataset containing over 250K images, 760K questions, and around 10M answers. [149] proposed a neural-based approach to answer the questions regarding the input

[11] A kind of long-term memory that includes conscious recall of previous activities together with their meaning.

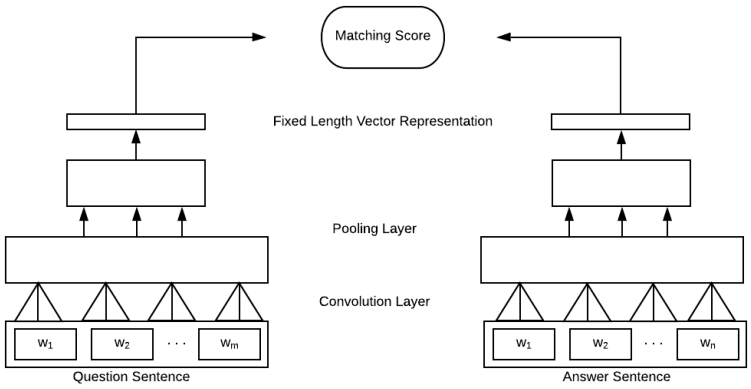

Fig. 13. Fixed length vector sentence representation for input Questions and Answers [145].

Input: Jane went to the hallway.
Input: Mary walked to the bathroom.
Input: Sandra went to the garden.
Input: Daniel went back to the garden.
Input: Sandra took the milk there.
Question: Where is the milk?
Answer: Garden
Input: It started boring, but then it got interesting.
Question: What's the sentiment?
Answer: Positive
Question: POS tags?
Answer: PRP VBD JJ, CC RB PRP VBD JJ.

Fig. 14. Example of Dynamic Memory Network (DMN) input-question-answer triplet

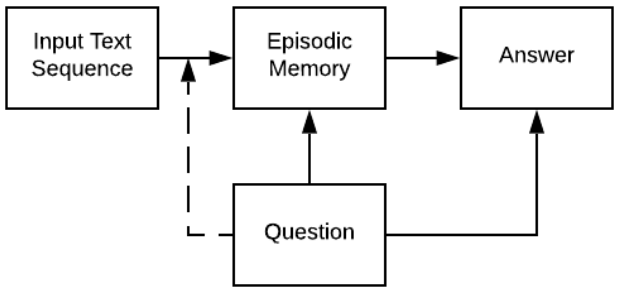

Fig. 15. Interaction between four modules of Dynamic Memory Network [68].

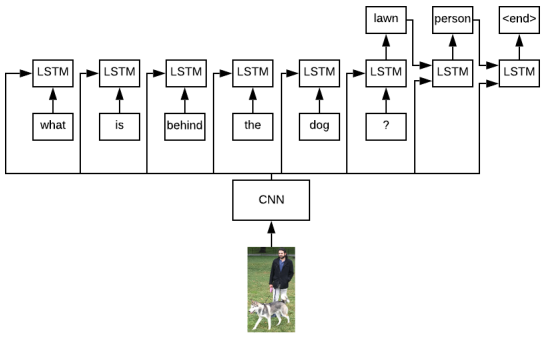

Fig. 16. Neural Image Question Answering [149].

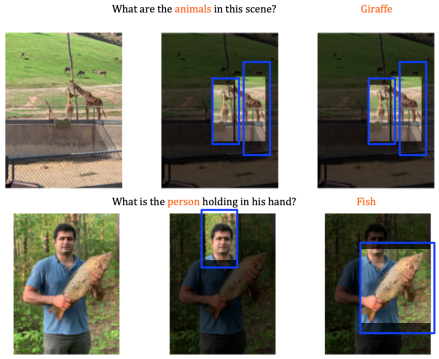

Fig. 17. Spatial Memory Network for VQA. Bright Areas are regions the model is attending [150].

images. As illustrated in Fig. 16, Neural-Image-QA is a deep network consisting of CNN and LSTM. Since the questions can have multiple answers, the problem is decomposed into predicting a set of answer words $a_{q,x} = \{a_1, a_2, ..., a_{N(q,x)}\}$ from a finite vocabulary set ν where $N(q, x)$ represents the count of answer words regarding a given question.

Do humans and computers look at the same regions to answer questions about an image? [151] tries to answer this question by conducting large-scale studies on *human attention* in VQA. Their findings show that VQAs do not seem to be looking at the same regions as humans. Finally, [150] incorporates a spatial memory network for VQA. Fig. 17 shows the inference process of their model. As illustrated in the figure, the specific attention mechanism in their system can highlight areas of interest in the input image.

G. Document Summarization

Document summarization refers to a set of problems involving generation of summary sentences given one or multiple documents as input.

Generally, text summarization fits into two categories:

1) **Extractive Summarization**, where the goal is to identify the most salient sentences in the document and return them as the summary.
2) **Abstractive Summarization**, where the goal is to generate summary sentences from scratch; they may contain novel words that do not appear in the original document.

Each of these methods has its own advantages and disadvantages. Extractive summarization is prone to generate long and sometimes overlapping summary sentences; however, the result reflects the author's mode of expression. Abstractive methods generate a shorter summary but they are hard to train.

There is a vast amount of research on the topic of text summarization using extractive and abstractive methods. As one of the earliest works on using neural networks for extractive summarization, [152] proposed a framework that used a ranking technique to extract the most salient sentences in the input. This model was improved by [153] which used a document-level encoder to represent sentences, and a classifier to rank these sentences. On the other hand, in abstractive summarization, it was [154] which, for the first time, used attention over a sequence-to-sequence (seq2seq) model for the problem of headline generation. However, since simple attention models perform worse than extractive models, therefore more effective attention models such as graph-based attention [155] and transformers [137] have been proposed for this task. To further improve abstractive text summarization models, [156] proposed the first pointer-generator model and applied it to the DeepMind QA dataset [157]. As a result of this work, the CNN/Daily Mail dataset emerged which is now one of the widely used datasets for the summarization task. A copy mechanism was also adopted by [158] for similar tasks. But their analysis reveals a key problem with attention-based encoder-decoder models: they often generate unusual summaries consisting of repeated phrases. Recently, [52] reached state-of-the-art results on the abstractive text summarization using a similar framework. They alleviated the unnatural summaries by avoiding generating unknown tokens and replacing these words with tokens from the input article. Later, researchers moved their focus to methods that use sentence-embedding to first select the most salient sentence in the document and then change them to make them more abstractive [159], [160]. In these models, salient sentences are extracted first and then a paraphrasing model is used to make them abstractive. The extraction employs a sentence classifier or ranker while the abstractor tries to remove the extra information in a sentence and present it as a shorter summary. Fast-RL [159] is the first framework in this family of works. In Fast-RL, the extractor is pre-trained to select salient sentences and the abstractor is pre-trained using a pointer-generator model to generate paraphrases. Finally, to merge these two non-differentiable components, they propose using Actor-Critic Q-learning methods in which the actor receives a single document and generates the output while the critic evaluates the output based on comparison with the ground-truth summary.

Though the standard way to evaluate the performance of summarization models is with ROUGE [57] and BLEU [58], there are major problems with such measures. For instance, the ROUGE measure focuses on the number of shared n-grams between two sentences. Such a method incorrectly assigns a low score to an abstractive summary that uses different words yet provides an excellent paraphrase that humans would rate highly. Clearly, better automated evaluation methods are needed in such cases.

There are additional problems with current summarization models. Shi *et al.* [161] provides a comprehensive survey on text summarization.

H. Dialogue Systems

Dialogue Systems are quickly becoming a principal instrument in human-computer interaction, due in part to their promising potential and commercial value. One application is automated customer service, supporting both online and bricks-and-mortar businesses. Customers expect an ever-increasing level of speed, accuracy, and respect while dealing with companies and their services. Due to the high cost of knowledgeable human resources, companies frequently turn to intelligent conversational machines. Note that the phrases *conversational machines* and *dialogue machines* are often used interchangeably.

Dialogue systems are usually *task-based* or *non-task-based* (Fig. 18). Though there might be Automatic Speech Recognition (ASR) and Language-to-Speech (L2S) components in a dialogue system, the discussion of this section is solely about the linguistic components of dialogue systems; concepts associated with speech technology are ignored.

Despite useful statistical models employed in the backend of dialogue systems (especially in language understanding modules), most deployed dialogue systems rely on expensive hand-crafted and manual features for operation. Furthermore, the generalizability of these manually engineered systems to other domains and functionalities is problematic. Hence, recent attention has focused on deep learning for the enhancement of performance, generalizability, and robustness. Deep learning facilitates the creation of end-to-end task-oriented dialogue systems, which enriches the framework to generalize conversations beyond annotated task-specific dialogue resources.

1) Task-based Systems: The structure of a task-based dialogue system usually consists of the following elements:

- *Natural Language Understanding (NLU)*: This component deals with understanding and interpreting user's spoken context by assigning a constituent structure to the spoken utterance (e.g., a sentence) and captures its syntactic representation and semantic interpretation, to allow the back-end operation/task. NLU is usually leveraged regardless of the dialogue context.
- *Dialogue Manager (DM)*: The generated representation by NLU would be handled by the dialogue manager, which investigates the context and returns a reasonable semantic-related response.
- *Natural Language Generation (NLG)*: The natural language generation (NLG) component produces an utterance based on the response provided by the DM component.

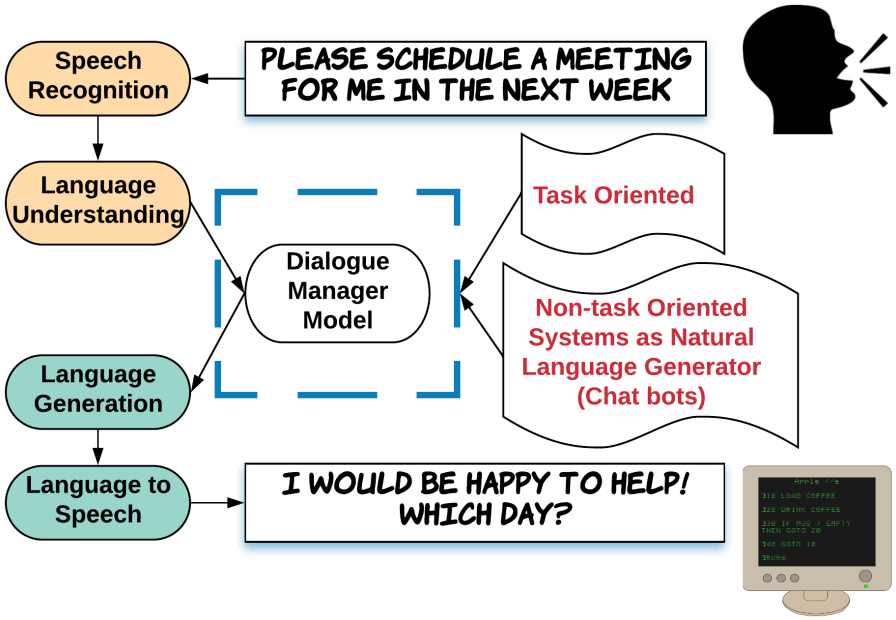

Fig. 18. The framework of a dialogue system. A dialogue system can be task oriented or used for natural language generation based on the user input which is also known as a chat bot.

The general pipeline is as follows: NLU module (i.e., semantic decoder) transforms the output of the speech recognition module to some dialogue elements. Then the DM processes these dialogue elements and provides a suitable response which is fed to the NLG for response generation. The main pipeline in NLU is to classify the user query domain and user intent, and fill a set of slots to create a semantic frame. It is usually customary to perform the intent prediction and the slot filling simultaneously [162]. Most of the task-oriented dialogue systems employ slot-filling approaches to classify user intent in the specific domain of the conversation. For this aim, having predefined tasks is required; this depends on manually crafted states with different associated slots. Henceforth, a designed dialogue system would be of limited or no use for other tasks.

Recent task-oriented dialogue systems have been designed based on deep reinforcement learning, which provided promising results regarding performance [163], domain adaptation [164], and dialogue generation [165]. This was due to a shift towards end-to-end trainable frameworks to design and deploy task-oriented dialogue systems. Instead of the traditionally utilized pipeline, an end-to-end framework incorporates and uses a single module that deals with external databases. Despite the tractability of *end-to-end dialogue systems* (i.e., easy to train and simple to engineer), due to their need for interoperability with external databases via queries, they are not well-suited for task-oriented settings. Some approaches to this challenge include converting the user input into internal representations [166], combining supervised and reinforced learning [167], and extending the memory network approach [168] for question-answering to a dialog system [169].

2) Non-task-based Systems: As opposed to task-based dialogue systems, the goal behind designing and deploying non-task-based dialogue systems is to empower a machine with the ability to have a natural conversation with humans [170]. Typically, chatbots are of one of the following types: *retrieval-based methods* and *generative methods*. Retrieval-based models have access to information resources and can provide more concise, fluent, and accurate responses. However, they are limited regarding the variety of responses they can provide due to their dependency on backend data resources. Generative models, on the other hand, have the advantage of being able to produce suitable responses when such responses are not in the corpus. However, as opposed to retrieval-based models, they are more prone to grammatical and conceptual mistakes arising from their generative models.

Retrieval-based methods select an appropriate response from the candidate responses. Therefore, the key element is the query-response operation. In general, this problem has been formulated as a search problem and uses IR techniques for task completion [171]. Retrieval-based methods usually employ either *Single-turn Response Matching* or *Multi-turn Response Matching*. In the first type, the current query (message) is solely used to select a suitable response [172]. The latter type takes the current message and previous utterances as the system input and retrieves a response based on the instant and temporal information. The model tries to choose a response which considers the whole context to guarantee conversation consistency. An LSTM-based model has been proposed [173] for context and response vectors creation. In [174], various features and multiple data inputs have been incorporated to be ingested using a deep learning framework. Current base models regarding retrieval-based chatbots rely on multi-turn

response selection augmented by an attention mechanism and sequence matching [175].

Generative models don't assume the availability of predefined responses. New responses are produced from scratch and are based on the trained model. Generative models are typically based on sequence to sequence models and map an input query to a target element as the response. In general, designing and implementing a dialogue agent to be able to converse at the human level is very challenging. The typical approach usually consists of learning and imitating human conversation. For this goal, the machine is generally trained on large corpora of conversations. However, this does not directly remedy the issue of *encountering out-of-corpus conversation*. The question is: *How can an agent be taught to generate proper responses to conversations that it never has seen?* It must handle content that is not exactly available in the data corpus that the machine has been trained on, due to the lack of content matching between the query and the corresponding response, resulting from the wide range of plausible queries that humans can provide.

To tackle the aforementioned general problem, some fundamental questions must be answered: **(1)** What are the core characteristics of a natural conversation? **(2)** How can these characteristics be measured? **(3)** How can we incorporate this knowledge in a machine, i.e., the dialogue system? Effective integration of these three elements determines the intelligence of a machine. A qualitative criterion is to observe if the generated utterances can be distinguished from natural human dialogues. For quantitative evaluation, adversarial evaluation was initially used for quality assessment of sentence generation [176] and employed for quality evaluation of dialogue systems [177]. Recent advancements in sequence to sequence modeling encouraged many research efforts regarding natural language generation [178]. Furthermore, deep reinforcement learning yields promising performance in natural language generation [165].

3) Final note on dialogue systems: Despite remarkable advancements in AI and much attention dedicated to dialogue systems, in reality, successful commercial tools, such as Apple's Siri and Amazon's Alexa, still heavily rely on handcrafted features. It still is very challenging to design and train data-driven dialogue machines given the complexity of the natural language, the difficulties in framework design, and the complex nature of available data sources.

VI. Conclusion

In this article, we presented a comprehensive survey of the most distinguished works in Natural Language Processing using deep learning. We provided a categorized context for introducing different NLP core concepts, aspects, and applications, and emphasized the most significant conducted research efforts in each associated category. Deep learning and NLP are two of the most rapidly developing research topics nowadays. Due to this rapid progress, it is hoped that soon, new effective models will supersede the current state-of-the-art approaches. This may cause some of the references provided in the survey to become dated, but those are likely to be cited by new publications that describe improved methods

Neverthless, one of the essential characteristics of this survey is its educational aspect, which provides a precise understanding of the critical elements of this field and explains the most notable research works. Hopefully, this survey will guide students and researchers with essential resources, both to learn what is necessary to know, and to advance further the integration of NLP with deep learning.

References

[1] C. D. Manning, C. D. Manning, and H. Schütze, *Foundations of statistical natural language processing*. MIT Press, 1999.

[2] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, pp. 649–657, 2015.

[3] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[4] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[5] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

[6] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1717–1724, 2014.

[7] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2107–2116, 2017.

[8] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, pp. 1764–1772, 2014.

[9] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," in *ICML*, pp. 173–182, 2016.

[10] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1818–1826, 2014.

[11] B. Plank, A. Søgaard, and Y. Goldberg, "Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss," *arXiv preprint arXiv:1604.05529*, 2016.

[12] C. D. Manning, "Part-of-speech tagging from 97% to 100%: is it time for some linguistics?," in *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 171–189, Springer, 2011.

[13] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[14] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *arXiv preprint arXiv:1511.08308*, 2015.

[15] J. Zhou and W. Xu, "End-to-end learning of semantic role labeling using recurrent neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 1127–1137, 2015.

[16] D. Marcheggiani, A. Frolov, and I. Titov, "A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling," *arXiv preprint arXiv:1701.02593*, 2017.

[17] L. He, K. Lee, M. Lewis, and L. Zettlemoyer, "Deep semantic role labeling: What works and what's next," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 473–483, 2017.

[18] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[19] T. Greenwald, "What exactly is artificial intelligence, anyway?." https://www.wsj.com/articles/what-exactly-is-artificial-intelligence-anyway-1525053960, April 2018. Wall Street Journal Online Article.

[20] U. Sivarajah, M. M. Kamal, Z. Irani, and V. Weerakkody, "Critical analysis of big data challenges and analytical methods," *Journal of Business Research*, vol. 70, pp. 263–286, 2017.

[21] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[22] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[23] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 129–136, 2011.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[25] C. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 69–78, 2014.

[26] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks," *arXiv preprint arXiv:1412.1058*, 2014.

[27] R. Johnson and T. Zhang, "Semi-supervised convolutional neural networks for text categorization via region embedding," in *Advances in neural information processing systems*, pp. 919–927, 2015.

[28] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2335–2344, 2014.

[29] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pp. 39–48, 2015.

[30] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[33] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[34] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in neural information processing systems*, pp. 2172–2180, 2016.

[35] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[36] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[37] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[38] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," *arXiv preprint arXiv:1701.06547*, 2017.

[39] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86, Association for Computational Linguistics, 2002.

[40] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[41] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb., pp. 1137–1155, 2003.

[42] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, pp. 1188–1196, 2014.

[43] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[44] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, pp. 3294–3302, 2015.

[45] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[46] G. Lebanon *et al.*, *Riemannian geometry and statistical machine learning*. LAP LAMBERT Academic Publishing, 2015.

[47] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive datasets*. Cambridge University Press, 2014.

[48] Y. Goldberg, "Neural network methods for natural language processing," *Synthesis Lectures on Human Language Technologies*, vol. 10, no. 1, pp. 1–309, 2017.

[49] J. Wehrmann, W. Becker, H. E. Cagnini, and R. C. Barros, "A character-based convolutional neural network for language-agnostic Twitter sentiment analysis," in *Neural Networks (IJCNN), 2017 International Joint Conference on*, pp. 2384–2391, IEEE, 2017.

[50] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[51] J. Botha and P. Blunsom, "Compositional morphology for word representations and language modelling," in *International Conference on Machine Learning*, pp. 1899–1907, 2014.

[52] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *ACL*, vol. 1, pp. 1073–1083, 2017.

[53] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," *arXiv preprint arXiv:1705.04304*, 2017.

[54] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, pp. 1171–1179, 2015.

[55] K. Goyal, G. Neubig, C. Dyer, and T. Berg-Kirkpatrick, "A continuous relaxation of beam search for end-to-end training of neural sequence models," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[56] W. Kool, H. Van Hoof, and M. Welling, "Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement," in *International Conference on Machine Learning*, pp. 3499–3508, 2019.

[57] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, pp. 74–81, 2004.

[58] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on Association for Computational Linguistics*, pp. 311–318, Association for Computational Linguistics, 2002.

[59] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72, 2005.

[60] Y. Keneshloo, T. Shi, C. K. Reddy, and N. Ramakrishnan, "Deep reinforcement learning for sequence to sequence models," *arXiv preprint arXiv:1805.09461*, 2018.

[61] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," *arXiv preprint arXiv:1511.06732*, 2015.

[62] W. Zaremba and I. Sutskever, "Reinforcement learning neural Turing machines-revised," *arXiv preprint arXiv:1505.00521*, 2015.

[63] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," in *Reinforcement Learning*, pp. 5–32, Springer, 1992.

[64] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[65] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[66] H. Daumé, J. Langford, and D. Marcu, "Search-based structured prediction," *Machine learning*, vol. 75, no. 3, pp. 297–325, 2009.

[67] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[68] V. Mnih, N. Heess, A. Graves, *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, pp. 2204–2212, 2014.

[69] C. Sun, L. Huang, and X. Qiu, "Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence," *arXiv preprint arXiv:1903.09588*, 2019.

[70] P. Resnik and J. Lin, "Evaluation of NLP systems," *The handbook of computational linguistics and natural language processing*, vol. 57, pp. 271–295, 2010.

[71] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, "Ask me anything: Dynamic memory networks for natural language processing," in *International Conference on Machine Learning*, pp. 1378–1387, 2016.

[72] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[73] D. Andor, C. Alberti, D. Weiss, A. Severyn, A. Presta, K. Ganchev, S. Petrov, and M. Collins, "Globally normalized transition-based neural networks," *arXiv preprint arXiv:1603.06042*, 2016.

[74] L. Liu, J. Shang, X. Ren, F. F. Xu, H. Gui, J. Peng, and J. Han, "Empower sequence labeling with task-aware neural language model," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[75] Z. Yang, R. Salakhutdinov, and W. W. Cohen, "Transfer learning for sequence tagging with hierarchical recurrent networks," *arXiv preprint arXiv:1703.06345*, 2017.

[76] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," *arXiv preprint arXiv:1603.01354*, 2016.

[77] M. Yasunaga, J. Kasai, and D. Radev, "Robust multilingual part-of-speech tagging via adversarial training," *arXiv preprint arXiv:1711.04903*, 2017.

[78] W. Ling, T. Luís, L. Marujo, R. F. Astudillo, S. Amir, C. Dyer, A. W. Black, and I. Trancoso, "Finding function in form: Compositional character models for open vocabulary word representation," *arXiv preprint arXiv:1508.02096*, 2015.

[79] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1638–1649, 2018.

[80] B. Bohnet, R. McDonald, G. Simoes, D. Andor, E. Pitler, and J. Maynez, "Morphosyntactic tagging with a Meta-BiLSTM model over context sensitive token encodings," *arXiv preprint arXiv:1805.08237*, 2018.

[81] J. Legrand and R. Collobert, "Joint RNN-based greedy parsing and word composition," *arXiv preprint arXiv:1412.7028*, 2014.

[82] J. Legrand and R. Collobert, "Deep neural networks for syntactic parsing of morphologically rich languages," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 573–578, 2016.

[83] A. Kuncoro, M. Ballesteros, L. Kong, C. Dyer, G. Neubig, and N. A. Smith, "What do recurrent neural network grammars learn about syntax?," *arXiv preprint arXiv:1611.05774*, 2016.

[84] J. Liu and Y. Zhang, "In-order transition-based constituent parsing," *arXiv preprint arXiv:1707.05000*, 2017.

[85] D. Fried, M. Stern, and D. Klein, "Improving neural parsing by disentangling model combination and reranking effects," *arXiv preprint arXiv:1707.03058*, 2017.

[86] N. Kitaev and D. Klein, "Constituency parsing with a self-attentive encoder," *arXiv preprint arXiv:1805.01052*, 2018.

[87] 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)*, pp. 740–750, 2014.

[88] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," *arXiv preprint arXiv:1611.01734*, 2016.

[89] E. Kiperwasser and Y. Goldberg, "Simple and accurate dependency parsing using bidirectional LSTM feature representations," *arXiv preprint arXiv:1603.04351*, 2016.

[90] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," *arXiv preprint arXiv:1505.08075*, 2015.

[91] Z. Tan, M. Wang, J. Xie, Y. Chen, and X. Shi, "Deep semantic role labeling with self-attention," *arXiv preprint arXiv:1712.01586*, 2017.

[92] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," *arXiv preprint arXiv:1703.04826*, 2017.

[93] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum, "Linguistically-informed self-attention for semantic role labeling," *arXiv preprint arXiv:1804.08199*, 2018.

[94] L. He, K. Lee, O. Levy, and L. Zettlemoyer, "Jointly predicting predicates and arguments in neural semantic role labeling," *arXiv preprint arXiv:1805.04787*, 2018.

[95] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[96] Z. Tan, M. Wang, J. Xie, Y. Chen, and X. Shi, "Deep semantic role labeling with self-attention," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[97] S. Pradhan, A. Moschitti, N. Xue, H. T. Ng, A. Björkelund, O. Uryupina, Y. Zhang, and Z. Zhong, "Towards robust linguistic analysis using OntoNotes," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 143–152, 2013.

[98] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[99] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward, "Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 4, pp. 694–707, 2016.

[100] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, 2016.

[101] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification.," in *AAAI*, vol. 333, pp. 2267–2273, 2015.

[102] C. Zhou, C. Sun, Z. Liu, and F. Lau, "A C-LSTM neural network for text classification," *arXiv preprint arXiv:1511.08630*, 2015.

[103] A. Conneau, H. Schwenk, L. Barrault, and Y. LeCun, "Very deep convolutional networks for text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, vol. 1, pp. 1107–1116, 2017.

[104] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 562–570, 2017.

[105] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using LSTM for region embeddings," *arXiv preprint arXiv:1602.02373*, 2016.

[106] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 328–339, 2018.

[107] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug., pp. 2493–2537, 2011.

[108] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding.," in *Interspeech*, pp. 3771–3775, 2013.

[109] F. Dernoncourt, J. Y. Lee, and P. Szolovits, "NeuroNER: an easy-to-use program for named-entity recognition based on neural networks," *Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2017.

[110] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pp. 142–147, Association for Computational Linguistics, 2003.

[111] K. Clark, M.-T. Luong, C. D. Manning, and Q. V. Le, "Semi-supervised sequence modeling with cross-view training," *arXiv preprint arXiv:1809.08370*, 2018.

[112] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[113] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pp. 1201–1211, Association for Computational Linguistics, 2012.

[114] K. Clark and C. D. Manning, "Deep reinforcement learning for mention-ranking coreference models," *arXiv preprint arXiv:1609.08667*, 2016.

[115] K. Lee, L. He, and L. Zettlemoyer, "Higher-order coreference resolution with coarse-to-fine inference," *arXiv preprint arXiv:1804.05392*, 2018.

[116] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 167–176, 2015.

[117] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[118] T. Nasukawa and J. Yi, "Sentiment analysis: Capturing favorability using natural language processing," in *Proceedings of the 2nd International Conference on Knowledge Capture*, pp. 70–77, ACM, 2003.

[119] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*, pp. 519–528, ACM, 2003.

[120] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1422–1432, 2015.

[121] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 513–520, 2011.

[122] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Proceedings of the conference on empirical methods in natural language processing*, pp. 151–161, Association for Computational Linguistics, 2011.

[123] X. Wang, Y. Liu, S. Chengjie, B. Wang, and X. Wang, "Predicting polarities of tweets by composing word embeddings with long short-term memory," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, pp. 1343–1353, 2015.

[124] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[125] Y. Wang, M. Huang, L. Zhao, *et al.*, "Attention-based LSTM for aspect-level sentiment classification," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 606–615, 2016.

[126] Y. Ma, H. Peng, T. Khan, E. Cambria, and A. Hussain, "Sentic lstm: a hybrid network for targeted aspect-based sentiment analysis," *Cognitive Computation*, vol. 10, no. 4, pp. 639–650, 2018.

[127] H. Xu, B. Liu, L. Shu, and P. S. Yu, "BERT post-training for review reading comprehension and aspect-based sentiment analysis," *arXiv preprint arXiv:1904.02232*, 2019.

[128] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Double embeddings and CNN-based sequence labeling for aspect extraction," *arXiv preprint arXiv:1805.04601*, 2018.

[129] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Prentice Hall, 2008.

[130] N. Kalchbrenner and P. Blunsom, "Recurrent continuous translation models," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1700–1709, 2013.

[131] L. E. Dostert, "The Georgetown-IBM experiment," *1955). Machine translation of languages. John Wiley & Sons, New York*, pp. 124–135, 1955.

[132] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[133] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[134] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

[135] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[136] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," *arXiv preprint arXiv:1705.03122*, 2017.

[137] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[138] K. Ahmed, N. S. Keskar, and R. Socher, "Weighted transformer network for machine translation," *arXiv preprint arXiv:1711.02132*, 2017.

[139] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *arXiv preprint arXiv:1803.02155*, 2018.

[140] S. Edunov, M. Ott, M. Auli, and D. Grangier, "Understanding back-translation at scale," *arXiv preprint arXiv:1808.09381*, 2018.

[141] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," *arXiv preprint arXiv:1406.3676*, 2014.

[142] B. F. Green Jr, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: an automatic question-answerer," in *Papers presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference*, pp. 219–224, ACM, 1961.

[143] A. Ittycheriah, M. Franz, W.-J. Zhu, A. Ratnaparkhi, and R. J. Mammone, "IBM's statistical question answering system.," in *TREC*, 2000.

[144] H. Cui, R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua, "Question answering passage retrieval using dependency relations," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 400–407, ACM, 2005.

[145] X. Qiu and X. Huang, "Convolutional neural tensor network architecture for community-based question answering.," in *IJCAI*, pp. 1305–1311, 2015.

[146] H. T. Ng, L. H. Teo, and J. L. P. Kwan, "A machine learning approach to answering questions for reading comprehension tests," 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*, pp. 124–132, Association for Computational Linguistics, 2000.

[147] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," *arXiv preprint arXiv:1611.01604*, 2016.

[148] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "VQA: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

[149] M. Malinowski, M. Rohrbach, and M. Fritz, "Ask your neurons: A neural-based approach to answering questions about images," in *Proceedings of the IEEE international conference on computer vision*, pp. 1–9, 2015.

[150] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *European Conference on Computer Vision*, pp. 451–466, Springer, 2016.

[151] A. Das, H. Agrawal, L. Zitnick, D. Parikh, and D. Batra, "Human attention in visual question answering: Do humans and deep networks look at the same regions?," *Computer Vision and Image Understanding*, vol. 163, pp. 90–100, 2017.

[152] R. Nallapati, F. Zhai, and B. Zhou, "SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents.," in *AAAI*, pp. 3075–3081, 2017.

[153] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking sentences for extractive summarization with reinforcement learning," in *NAACL:HLT*, vol. 1, pp. 1747–1759, 2018.

[154] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *EMNLP*, 2015.

[155] J. Tan, X. Wan, and J. Xiao, "Abstractive document summarization with a graph-based attentional neural model," in *ACL*, vol. 1, pp. 1171–1181, 2017.

[156] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 280–290, 2016.

[157] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *NIPS*, pp. 1693–1701, 2015.

[158] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *ACL*, vol. 1, pp. 1631–1640, 2016.

[159] Y.-C. Chen and M. Bansal, "Fast abstractive summarization with reinforce-selected sentence rewriting," in *ACL*, 2018.

[160] Q. Zhou, N. Yang, F. Wei, S. Huang, M. Zhou, and T. Zhao, "Neural document summarization by jointly learning to score and select sentences," in *ACL*, pp. 654–663, ACL, 2018.

[161] T. Shi, Y. Keneshloo, N. Ramakrishnan, and C. K. Reddy, "Neural abstractive text summarization with sequence-to-sequence models," *arXiv preprint arXiv:1812.02303*, 2018.

[162] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, and Y.-Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM," in *Interspeech*, pp. 715–719, 2016.

[163] C. Toxtli, J. Cranshaw, *et al.*, "Understanding chatbot-mediated task management," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, p. 58, ACM, 2018.

[164] V. Ilievski, C. Musat, A. Hossmann, and M. Baeriswyl, "Goal-oriented chatbot dialog management bootstrapping with transfer learning," *arXiv preprint arXiv:1802.00500*, 2018.

[165] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep reinforcement learning for dialogue generation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1192–1202, 2016.

[166] T.-H. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," *arXiv preprint arXiv:1604.04562*, 2016.

[167] J. D. Williams and G. Zweig, "End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning," *arXiv preprint arXiv:1606.01269*, 2016.

[168] S. Sukhbaatar, J. Weston, R. Fergus, *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, pp. 2440–2448, 2015.

[169] A. Bordes, Y.-L. Boureau, and J. Weston, "Learning end-to-end goal-oriented dialog," *arXiv preprint arXiv:1605.07683*, 2016.

[170] A. Ritter, C. Cherry, and W. B. Dolan, "Data-driven response generation in social media," in *Proceedings of the conference on empirical methods in natural language processing*, pp. 583–593, Association for Computational Linguistics, 2011.

[171] Z. Ji, Z. Lu, and H. Li, "An information retrieval approach to short text conversation," *arXiv preprint arXiv:1408.6988*, 2014.

[172] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Advances in neural information processing systems*, pp. 2042–2050, 2014.

[173] R. Lowe, N. Pow, I. Serban, and J. Pineau, "The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems," *arXiv preprint arXiv:1506.08909*, 2015.

[174] R. Yan, Y. Song, and H. Wu, "Learning to respond with deep neural networks for retrieval-based human-computer conversation system," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 55–64, ACM, 2016.

[175] X. Zhou, L. Li, D. Dong, Y. Liu, Y. Chen, W. X. Zhao, D. Yu, and H. Wu, "Multi-turn response selection for chatbots with deep attention matching network," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, pp. 1118–1127, 2018.

[176] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[177] A. Kannan and O. Vinyals, "Adversarial evaluation of dialogue models," *arXiv preprint arXiv:1701.08198*, 2017.

[178] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.