A Brief Survey into the Evolution of Machine Translation

Arjun Karuvally *
akaruvally@cs.umass.edu

Rakesh Radhakrishnan Menon *
rrmenon@cs.umass.edu

Abstract

In this paper, we perform a brief survey on the task of machine translation where given a sentence(s) in one natural language we train a model to predict the same sentence(s) in another natural language. To this end, we examine methods from classical natural language processing such as rule-based machine translation models and statistical machine translation models to some of the recently developed state-of-the art neural machine translational systems. We also talk about some of the existing datasets and the metrics that are used for evaluating current machine translation models.

1 Introduction

Machine translation (MT) is a natural language task where we train a model to automatically convert text/speech between two human languages. The MT task usually takes as input a "source" language and produces an output translation in a "target" language. While this may seem trivial at first as a simple word-to-word translation from one language to another, the real challenge lies in the ability to produce a translation that is correct morphologically, syntactically and semantically. This further emphasizes the need for a model to understand the complete meaning of the text in both the source and target languages.

As a direct consequence of the above fact, we can come to the conclusion that representation of the language plays a crucial role in the advancement of machine translation. This leads to the discussion about the importance of language models for natural language understanding. In this paper however, we do not emphasize on the literature from language models. Our focus is on the machine translation task and the survey of methods that have been developed for performing the task. The interested reader can however refer to (Rosenfeld, 1996; Bengio et al., 2003; Mikolov et al., 2010; Pennington et al., 2014; Mikolov et al., 2013) for some background on language models.

In this paper, we first discuss about different approaches to machine translation namely, rule-based methods and statistical methods in addition to the merits and demerits of each approach (Section 2). Next, we delve deeper into statistical machine translation and look at some of the classical approaches that have been adopted in the past (Section 4.1 and Section 4.2). Further, we discuss more modern neural statistical machine translation models which has seen a resurgence due to the vast amounts of improvement in deep learning research (LeCun et al., 2015) (Section 4.3). Finally, we discuss some of the current datasets and evaluation metrics that are used for evaluating the current models in machine translation.

We would like to note that this paper is by no measure an exhaustive review of machine translation techniques in the past. Through this paper, we wish to highlight some approaches for machine translation taken to bring a novice reader up to pace in machine translation with some well-defined methods from the past and some of the state-of-the-art neural machine translation models in the present.

2 Approaches to Machine Translation

Machine Translation has been there in literature since the development of computers. Many leading researchers at the time realized the requirement of automatic translation and its importance in the future of natural language processing. Ini-

*Equal Contribution. Names arranged in order of first names.

tial approaches to machine translation include incorporation of linguistic knowledge giving rise to Rule-based machine translation. Further down the line, with the incorporation of neural networks in the design of machine translation systems, the methods evolved beyond the explicit inclusion of linguistic knowledge to statistical models that understand the complicated structure of language and predict its translation.

2.1 Rule-based Machine Translation (RBMT)

Before machines became faster and enabled us to perform complicated calculations in a short time, machine translation was mainly focused on rule based systems. Complicated set of rules were explicitly coded into systems to achieve reasonable translation accuracy. The rules were based on linguistic knowledge of the source and target language the system is required to work on. When an input source sentence is provided, the system produces output based on morphological, syntactic and semantic analysis of the source and target languages involved in the translation task.
There are different types of RBMT. Direct RBMT systems directly map the source to target using dictionaries and simple rules. The interlingual RBMT uses an abstract meaning to achieve the translation. The most widely used form of RBMT is the transfer based system. In this model, there are three stages - analysis, transfer and generation. The analysis phase analyses the source sentence to determine the grammatical structure of it. The transfer phase converts the analysis in the source sentence to a suitable form in the target language based on rules that define the conversion. The final part is generation phase that generates the target sentence from the representation from the transfer phase. RBMT first analyses the input text for morphology, syntax and semantics based on rules coded from linguistic knowledge of the source language. This analyzed knowledge is then converted into an interlingua representation that is a common representation among languages which is then converted into the target language based on the information coded from linguistic knowledge of the target language. It is important to note here that heavy use of linguistics is involved in the model which results in the reduction in popularity of the model in later years. A general RBMT system contains several components like the Source Language (SL) morphological analyzer, SL parser that is a syntax analyzer for source language sentences, translator, Target Language (TL) morphological generator, TL parser and many dictionaries. The dictionaries are used for each of these conversions. Although RBMT systems are not very popular now, they have significant advantages over statistical systems which includes ease of debugging, easy inclusion of rules and dealing with edge cases. Statistical systems tend to omit rare cases due to less loss associated with the wrong prediction on that. On the other hand, they have significant disadvantages too which includes maintaining dictionaries and adaption to change in grammatical structure over time and the requirement of high amount of workforce to create and maintain such a system. In the context of RBMT, a popular open source RBMT system called Apertium (Forcada et al., 2011) exist which is a shallow transfer machine translation system. It uses finite state transducers for its lexical transformations and hidden markov models for POS tagging and word category disambiguation.
Interested readers should consider (Hutchins, 1986) which is an old but detailed book on the approaches used in rule based machine translation systems.

2.2 Statistical Machine Translation (SMT)

The use of statistical methods was first suggested by Warren Weaver in July 1949 (Weaver, 1949) in a memorandum to some of his acquaintances. The memorandum mentions the possibility of using the then recently invented digital computers for machine translation. This memorandum is on of the most influential publication in machine translation as it had envisioned goals and methods before most people had any idea on what computers would be capable of. He mentions in the memorandum that the use of single words to translate a sentence will not be fruitful and for a system to be able to translate well, a window of N words have to be taken from either sides of the word to understand its context. His second proposal was based on the assumption that there are logical elements in language and that the problem of machine translation could be solved formally using mathematical methods. His third proposal was to use cryptographic methods, which was inspired from a wartime experience of deciphering a Turkish text which without even knowing the lan-

guage the mathematician was able to solve this. The ideas of cryptography was linked to information theory which was being developed by Shannon(Shannon and Weaver, 1949) at the time. The memorandum spurred some research in the area of machine translation based on statistical properties of languages and became a milestone in machine translation literature.

The approach of using statistical methods for machine translation was initially abandoned due to the unavailability of fast computers at the time and the impracticality of encoding the source texts at the time into a form readable by machines and theoretical objections. As machines became faster, this line of research was reignited with a paper demonstrating the use of statistical methods for machine translation(Brown et al., 1990). In this paper, the author considers the task of single sentence translation. The work is based on the assumption that every sentence in one language is a translation of some sentences in the other. The probability that a particular sentence T in one language is a translation of another sentence S in the source language is denoted as $p(T|S)$. The problem of machine translation is then formulated as "Given a sentence T in the target language, we seek the sentence S from which translation produced T". Mathematically, it can be stated as choosing S that maximizes the product $p(S)p(T|S)$. The authors, additionally introduced the concept of alignment(Figure 1), where a word in the source sentence is said to be aligned with a word in the target when that word influences the production of the word in the target. This is one of the important concepts of statistical translation that is introduced in the paper which would be used for many state of the art approaches in later works. The next objective of the system would

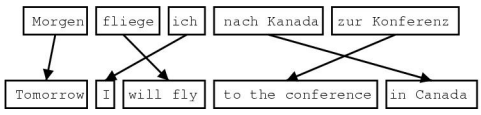

Figure 1: An example of word alignment.

then be the search of source sentence S that maximizes $p(S)p(T|S)$. This is a difficult objective to tackle as the search space is huge with many sentences having similar meaning. To achieve this, suboptimal search strategies are generally implemented. (Brown et al., 1990) uses a variant of the stack search. In the literature, beam search is usually encountered as a search strategy, which is a suboptimal version of the BFS(Breadth First Search) algorithm. It has a parameter beam width that indicate how many promising nodes are to be searched further. Beam width of infinity indicate that the graph is completely traversed. If beam width is very narrow, many branches that may be promising in the future may be ignored. So beam width is an additional hyperparameter that is to be tuned according to problem. The next few sections deal with statistical machine translation methods in much more detail.

3 Evaluation Metrics

Metrics that define how good a result an algorithm has produced is required for any statistical judgment between two systems/algorithms and so we have added this section before discussing any statistical MT models in order to acquaint the reader with some of the evaluation metrics used in machine translation. In machine translation, we have two metrics that have been developed in recent times: (a) BLEU (Papineni et al., 2002) and (b) chrF (Popović, 2015). Of the two, BLEU seems to be the most popular choice as it has been part of the literature for a long period of time. Most papers that date before the introduction of BLEU never report any statistical quantification for how good a translation the system has produced but instead report some qualitative results with some examples. In this paper, we discuss NMT papers that use at least one of the two metrics that have been proposed.

3.1 BLEU

BLEU metric, developed by (Papineni et al., 2002), is a measure of the quality of the text that has been produced by an MT system. The metric gives a score which compares the n-grams obtained from each of the candidate sentences to some human(good quality) reference sentences translations. Usually, the unigram probability is calculated to measure the quality of translation. To this end, the metric will calculate the number of matching words with the reference sentence and counts the number of matches. The final output is a probability of the number of matches over the total number of words. It should also be noted that the matches are position-independent.

The BLEU score also introduces a modified n-gram precision, where the maximum number of

occurrences of a word in the reference sentences is taken into account, and clips the maximum possible number of occurrences of the same word in the candidate sentences. Furthermore, the length of a sentence is penalized by the use of *brevity penalty* factor. Whenever the length of the candidate sentences is less than or equal to the maximum length of the reference sentences, then penalty factor is kept at 1. Combining all these factors, the final BLEU score takes the weighted geometric mean of the precision scores $\mathbf{p}_n$, weighted by positive weights $\mathbf{w}_n$, of the test corpus and the result is multiplied by the exponential *brevity penalty* factor.

So, putting together all the pieces for BLEU, we have for c as length of translation and r as effective length of reference corpus,

$$\mathbf{p}_n = \frac{\sum_{\mathbb{C} \in \text{Candidates}} \sum_{\text{n-gram} \in \mathbb{C}} \text{count}_{clip}(\text{n-gram})}{\sum_{\mathbb{C}' \in \text{Candidates}} \sum_{\text{n-gram} \in \mathbb{C}'} \text{count}(\text{n-gram})}$$

$$\text{BP} = \begin{cases} 1 & c > r \\ e^{(1-\frac{r}{c})} & c \leq r \end{cases}$$

$$\text{BLEU} = \text{BP} \cdot \exp(\sum_{n=1}^{N} \mathbf{w}_n \log \mathbf{p}_n)$$

3.2 chrF

While the BLEU score was able to provide a reasonable metric for machine translation, there was some amount of disparity in the relation between the BLEU score and human-evaluated scores. So there was a need to introduce a measure that most likely correlated with human level judgments. Thus the chrF measure (Popović, 2015) was introduced which calculates an F-measure based on character n-grams. The general formula for chrF is given as,

$$\text{chrF}\beta = (1 + \beta^2) \frac{\text{CHRP.CHRR}}{\beta.\text{CHRP} + \text{CHRR}}$$

where CHRP (precision score) is the percentage of n-grams in the hypothesis which have a counterpart in the reference while CHRR (recall score) is the percentage of n-grams which are in the reference and also present in the hypothesis. β is a parameter that controls the amount of extra weight we put on the recall score.

4 An Overview of Approaches in SMT

SMT is an umbrella term for many methods that use statistical methods for the task of machine translation. There comprise of different categories of statistical methods which include methods that involve explicit feature engineering like Phrase-based MT, Syntax-based MT and some like Neural MT that is able to directly work on top of raw data.

4.1 Phrase-based MT

In these type of models, phrase is the atomic unit in translation(Och et al., 1999). This is a very powerful model since a phrase is able to encode information about the context that is useful for translation. This is used along with alignment to translate sentences into the target language. The main component of this model is the Phrase Translation Tables which contain phrases, their translations and the probability of the translation. It can be noticed here that the phrases may not be linguistic phrases but any sequence of words. It is also seen from experiments that usage of only linguistic phrases results in a decrease in performance(Koehn et al., 2003). Phrase-based MT are powerful model and it was used in Google Translate even though it has now shifted to Neural Machine Translation model(Wu et al., 2016). Another popular approach to phrase-based machine translation is the use of a joint probability statistical model(Marcu and Wong, 2002). This involves the use of noise channel framework where each source sentence in a parallel corpus is assumed to generate a target sentence using a stochastic process and the parameters of the model.

4.2 Syntax-based MT

Another interesting model is the Syntax-based MT (Yamada and Knight, 2001) which uses statistical properties and syntax properties of language to achieve the task of translation. The idea is to get the best out of both worlds. Statistical methods had the issue that sometimes the translation did not follow the syntax of the target language. Syntax properties can be incorporated to fix this aspect of statistical machine translation. To this end, a syntax tree is processed using a syntactic parser for the input source language sentence and the syntax tree is then used by to create model a system for machine translation using statistical methods. Operations include reordering, insertion and translation of leaf nodes. Finally, the output obtained is in the form of string from this parse tree. A didactic example showing the operations involved in syntax based statistical MT has been outlined in Figure 2.

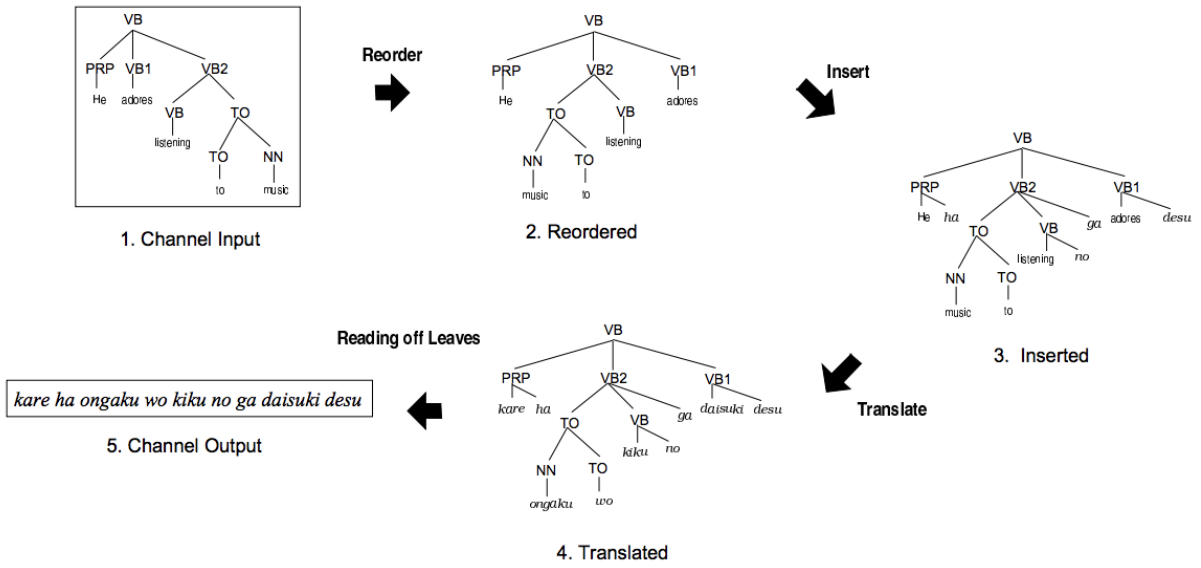

Figure 2: Syntax-based MT example. From (Yamada and Knight, 2001).

In the following years there was a rise in the amount of data available for training but syntax-based and pure phrase-based machine translations required a lot of preprocessing and feature construction which became increasingly difficult. Coupled with the improvement in the processing power of computer systems, there was an increasing amount of research in the area of Deep Learning which catalyzed the shift to Neural Network based Machine Translation Models which is covered in detail in the next few sections.

4.3 Neural MT (NMT)

A third popular mechanism for performing statistical machine translation is through the use of neural networks. While the underlying natural language techniques do not vary much in these models, neural networks are able to supplement existing models through their increased representative power for language modeling.

4.3.1 Recursive Hetero-Associative Memory for Machine Translation

One of the first neural machine translation models to be proposed was the Recursive Hetero-Associative Memory (RHAM) architecture (Forcada and Ñeco, 1997). The architecture, which is a derivative of a previously proposed RAAM architecture (Pollack, 1990), used an encoder-decoder model for converting the symbols from one language to another. Taking a slight detour from the notations in the original paper, the input to this encoder is a concatenation of representation $u_t(\in \mathbb{R}^S, t \geq 0)$ for the symbol in the source language, a state representation $r_t(\in \mathbb{R}^K, t \geq 0)$ and a bias term. The output of the encoder is a representation in the space $\mathbb{R}^K$ which is used as the state representation for the state for $r_t, t \geq 1$. The initial input to the encoder is u_0 is the representation of the first symbol and the state r_0 is initialized to zero for the first symbol. The output state from the encoder is used as the input state r_t for all subsequent time steps $t \geq 1$ until the end of the sequence of symbols in the first language. Mathematically, the new r_t can be represented as,

$$r_t = g(r_{t-1}, u_{t-1}), t \geq 1$$

where g() is the sigmoidal activation function. The decoder follows a similar architecture as the encoder except in the reverse direction where a encoded representation in $\mathbb{R}^T$ is provided as input and the output is a representation in $\mathbb{R}^{S'+T}$ where the first S' bits represents the encoding of the symbol in the target language and the next T bits represents the rest of the state r_t. Once again, this process is repeated until the end of the sequence in the target language is detected. The model is trained end-to-end using stochastic gradient descent approaches and was shown to perform well on tasks performed by the Mealy and deterministic generalized sequential machines. The training of this model resembles to some of the training of the more modern recurrent neural network architectures being used today and

perhaps laid the foundation for the use of more sequence-to-sequence models.

Although, neural networks were able to achieve this feat in the 1990s, one of the main bottlenecks for using neural network modules was the amount of hardware that was available at that time and the computational power needed to train these modules. However, through some recent improvements in hardware as well as deep learning research (LeCun et al., 2015), neural networks models have become a lot more feasible to train. In the past decade, we have already witnessed some of the major successes of deep learning combined with language modeling (Mikolov et al., 2010; Pennington et al., 2014; Mikolov et al., 2013), computer vision (Krizhevsky et al., 2012; Goodfellow et al., 2014) and reinforcement learning (Silver et al., 2016; Mnih et al., 2015). This leads to the natural question as to how helpful neural network models are for machine translation. The answer to this question lies in the next few sections of our paper which describe how neural network models, especially sequence-to-sequence models, have come to improve performance on machine translation. Due to the large volume of research papers that have come out using sequence-to-sequence models for neural machine translation, we have developed two different hierarchies to talk about the different architectures that have been proposed towards building state-of-the-art translation systems. The first hierarchy (Section 5) is based on the use of pure encoder-decoder models and models with attention(or alignment) mechanisms while the second hierarchy (Section 6) builds on the various kinds of inputs that can be provided to the machine translation system to produce better sentences in the target language(s).

5 Sequence-to-Sequence Models in NMT

The performance of an MT system relies heavily upon the ability of the model to understand the sentence that has been provided in the source language. So, as we have mentioned before, good language models are required for understanding the underlying meaning of the source sentence or at least the syntax and semantics of the sentence. With respect to neural network models, sequence-to-sequence models such as recurrent neural networks(RNNs) and long short term memory networks (LSTMs) make more for modeling language as the entire meaning of the sentence can be encoded in the hidden state of the chosen sequence-to-sequence models. In the next few subsections, we examine the usage of such sequence-to-sequence models as encoder-decoder and attention-based models for performing translation.

5.1 Encoder-Decoder Models

In this type of sequence to sequence models, the assumption is that the sentence to be translated can be encoded into a latent space that contains all the information conveyed by the sentence. The encoder performs the conversion into a latent representation while the decoder takes this latent space representation and converts it into the target sentence.

In (Cho et al., 2014), the idea of using RNN for the task of machine translation was proposed called the RNN Encoder-Decoder Model. In the paper, one RNN encodes the sequence into a fixed length vector representation, and the other RNN decodes the fixed length sequence into the target sentence. Both the encoder and decoder are jointly trained end-to-end to maximize the conditional distribution of the target words given the source sentence.

A recurrent neural network is a network that consists of a hidden state h and an optional output y which operates on a variable length input $\mathbf{x} =< x_1, x_2, \ldots, x_n >$. At each time step t, the update to the hidden state is given by,

$$h_t = f(h_{t-1}, x_t)$$

where f is a non-linear activation function. By training on the next symbol prediction task, RNNs are able to model the probability distribution over a sequence. Here the output at timestep t would be the conditional probability of the next word given the sequence seen so far, $p(x_t|x_{1:t-1})$. Using this conditional probability distribution, the probability of a sequence can be computed using a softmax function that converts scores into a probability distribution. The probability of sequence is then given by,

$$p(x) = \prod_{t=1}^{t=n} p(x_t|x_{1:t-1})$$

RNN based encoder-decoder network learns the conditional distribution of the target sequence

given the source sequence. The encoder of the network is straightforward in the sense that the variable length sequence is recursively input into the network to get a fixed length latent representation. The decoder varies from a traditional RNN in that both the hidden state and output are conditioned on the previous hidden state (h_{t-1}), the previous output (y_{t-1}) and the learned fixed length latent representation (c). Thus, the hidden state at time t is computed using

$$h_t = f(h_{t-1}, y_{t-1}, c)$$

The output from the network is passed through a softmax function to obtain the conditional probability distribution of the next word, which is given as

$$p(y_t|y_{1:t-1}, c) = p(h_t, y_{t-1}, c)$$

The components of the network are jointly trained to maximize the conditional distribution of the target sequence

$$\max_\theta \frac{1}{N} \sum_{n=1}^{N} \log p_\theta(\mathbf{y}_n|\mathbf{x}_n)$$

where θ is the model parameters, each $\mathbf{x}_n$ is the source language sequence and $\mathbf{y}_n$ is the generated target language sequence. The RNN in the paper

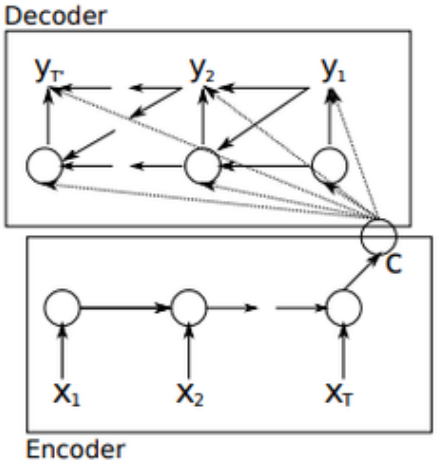

Figure 3: The RNN encoder-decoder architecture as shown in in (Cho et al., 2014).

uses a simplification of the LSTM network to allow the ability to adaptively remember and forget. The hidden unit they propose has a reset gate r_j that is computed as

$$r_j = \sigma([W_r x]_j + [U_z h_{t-1}]_j)$$

where σ is the sigmoid function and $[.]_j$ represents the j^{th} element of the matrix. W_r and U_z are the weight matrices that are learned by the network. The update gate z_j is computed by

$$h_j^t = z_j h_j^{t-1} + (1 - z_j) H_j^t$$

where H is computed as,

$$H_j^t = \Phi([Wx]_j + [U_z(r) \odot h_{t-1}]_j)$$

here it can be observed that when reset gate is close to zero, the term $U_z(r) \odot h_{t-1}$ is close to zero and the previous hidden state is "forgotten". This enables the network to choose to remember or forget any context at any time in the future allowing a compressed representation. On the other hand, the reset gate has the ability to control how much information is passed forward in the network allowing the network to learn long-term information.

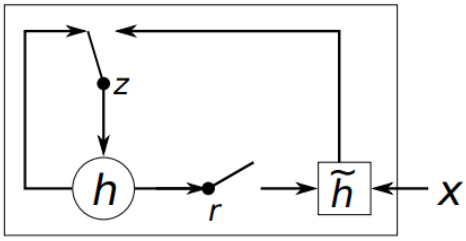

Figure 4: The architecture of the proposed gate as shown in (Cho et al., 2014), where h in the hidden representation. r is the reset gate that can choose to forget any information at a given point. z is the update gate that controls how much information is passed through

The evaluations are done in the English/French translation task of the WMT'14. The experiments show a considerable improvement in the BLEU scores from the baseline which is a phrase-based SMT system. The qualitative analysis of the RNN network shows that the model was able to learn meaningful embeddings where semantically similar words appear together (see Figure 5).

From the discussions in the paper, it is quite evident that using a recurrent neural network can learn the task of machine translation end-to-end and without any feature engineering. The simple system was able to outperform the baseline system even though it did not have much handcrafted features that it could use. This work spurred many works on the use of neural networks for the task

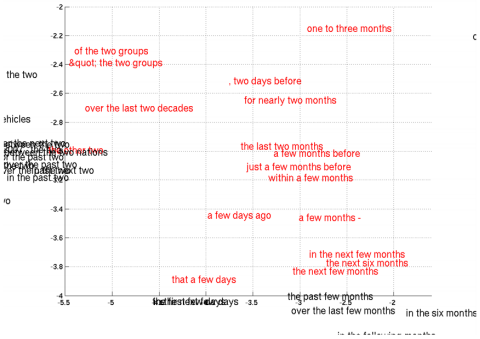

Figure 5: The embeddings learned by the RNN model that shows semantically similar words appearing together. Figure adapted from (Cho et al., 2014)

of machine translation. In view of the success of the RNN model in the task of machine translation, the discussion would be incomplete without the discussion of the use of LSTM for sequence-to-sequence modeling which gained popularity quickly and is also used for other sequence-to-sequence tasks. The work in (Sutskever et al., 2014) further goes on to show that LSTM networks trained end-to-end can considerably improve the BLEU score when compared to the gated RNN model.

Although RNN and LSTM networks are able to learn sequence-to-sequence models with good accuracy, they have a fundamental flaw in that they cannot capture information of long sentences within the latent representation. This leads to our next sub-section on attention models where we talk about works that deal with this issue in particular.

5.2 Attention-based Models

In the previous subsection, we have seen how using encoder decoder models are not so useful while trying to predict the translations for longer sentences. This is because the RNN/LSTM is not able to hold the entire information that is present in the sentence within its embedding. Put in other words, while decoding the state information is not able to provide the relevant information that is required for performing translation at a particular location in the target language. One possible method to overcome this problem, is to have the network be able to focus on relevant parts of the sequence in the source sentence to perform translation. The corresponding solution in deep learning comes in the form of an attention mechanism (also known as an alignment model), where the network decides how relevant each part of the information in a sequence-to-sequence model based on a similarity measure which could be cosine similarity, L2 distance or even a score provided by a neural network.

One of the first such models based on alignment was introduced in (Bahdanau et al., 2015). In this paper, the author(s) uses a bidirectional LSTM to encode the source sentences as done in any encoder-decoder model. For the decoder segment, the network compares all the hidden states of the input sentences (h_i's) with the current hidden state of the decoder (s_j) in order to produce an unnormalized score e_{ij}. The unnormalized score in this paper is computed using a feedforward neural network which takes as input h_i and s_j. Or in other words, $e_{ij} = a(h_i, s_j)$, where a is the function represented by the feedforward network. Here i is the index of the input hidden state and j is the index of the target hidden state. Following this, the scores are normalized using a softmax operation to produce probability scores $\alpha_i j$ for each hidden state in the input sequence. The context c_j that is hence required for decoding the word j in the target language is computed as $c_j = \sum_{i=1}^{T} \alpha_{ij} \times h_i$, where T is assumed to be the length of the source sentence. Hence, we can write the equation used for the model predicting the j^{th} word in the target sequence as,

$$p(y_j|y_{j-1}, y_{j-2}, \ldots, y_0, \mathbf{x}) = g(y_{j-1}, s_j, c_j)$$

where y_j indicates the target word y at the j^{th} position, $\mathbf{x}$ represents the entire input source sequence. Results on the WMT 14 dataset shows a marked improvement in the BLEU score of about 4-7 points over the baseline encoder-decoder models in (Cho et al., 2014). However, a more important result of the paper was the effect of longer sequences in attention models. As shown in Figure 6, attentional models are able to generate longer sentences better because of its ability to weight the context in the input hidden state. An additional insight provided by the author(s) is the alignment of words between English and French which clearly indicates how attention models assign high weights to the required context (see Figure 7).

While the alignment model proposed in (Bahdanau et al., 2015) was able to show how attention can help in providing weighted context for performing better translation, a major drawback with

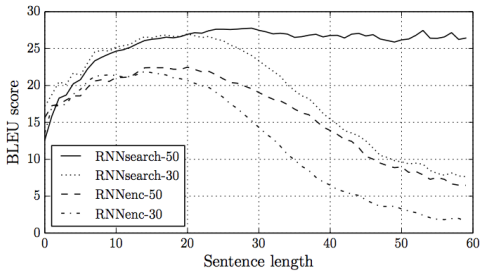

Figure 6: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. Figure taken from (Bahdanau et al., 2015).

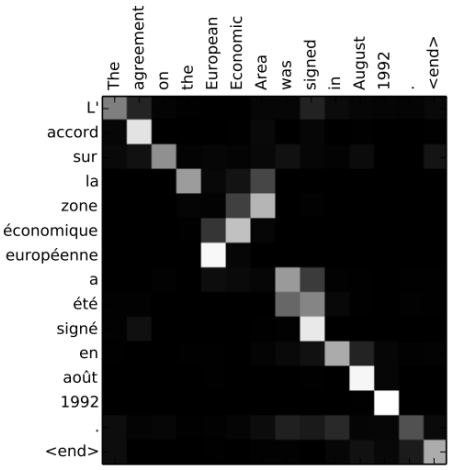

Figure 7: Alignment scores for English(column) to French(row) translation. Figure taken from (Bahdanau et al., 2015).

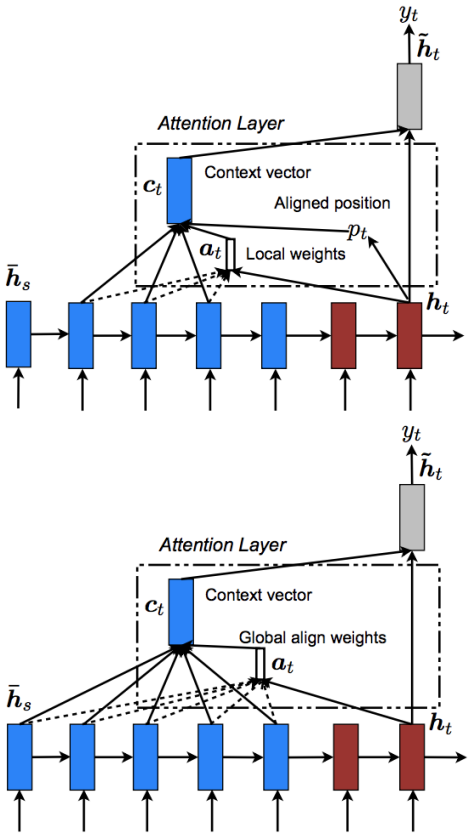

Figure 8: Local and Global attention models proposed in (Luong et al., 2015).

the approach was how the number of computations required for obtaining the context increased with the length of the input sequence. This would make the translation systems really slow as we try to decode some very large sentences. A slightly better solution to the above problem is to fix the context window over which the attention is performed. This is exactly what the author(s) propose in (Luong et al., 2015) with global attention and local attention models. While the author(s) do claim their global attention model to be a variant which is different from the model proposed in (Bahdanau et al., 2015), we do not go into further discussion about the model because of the lack of differences from a natural language perspective. However, for the sake of completeness of the paper, we would like to mention that while in (Bahdanau et al., 2015) the author(s) use a concatenated representation of the hidden states from the bi-directional LSTM, the global attention model in (Luong et al., 2015) uses the hidden representation from the final layer of a stacked LSTM architecture as shown in Figure 8. On the other hand, the local attention model can be seen as a blend between soft attention models, where some weight is applied all across the input sequence (also seen in global attention models), and hard attention models, where a particular hidden state is chosen to provide context for target word generation. The key difference however in local attention models is that they focus on only parts of the input sequence and the alignment function is completely differentiable unlike the case of hard attention. The advantage of this is that the models are now computationally less expensive and training happens a lot faster since we are not concerned with the entire input sequence. The slight disadvantage of these models is the error in judgment that can result from not having enough context to perform the translation. Having mentioned the differences, advantages and disadvantages of the local attention models, we dive right into describing the inner workings of it.

The local attention model first generates a

position p_t in the input sequence around which the local attention is performed. Following this the context vector is computed using the terms from $[p_t - D : p_t + D]$, where D is the window size indicating the number of words around p_t that are chosen to perform the alignment calculation. The window size D is a hyperparameter which the author(s) claim to set using empirical results. The computation for the context vector having already calculated the window of words to perform alignment over is quite different from the proposed function in (Bahdanau et al., 2015). In this paper instead the author(s) calculate an unnormalized score using some matrix operation and further normalize the scores to a softmax distribution to give alignment weights α_is. Now, a natural question to ask is, "how do we set p_t?". To do this, the author(s) propose two alternatives. The first alternative is to set $p_t = t$, that is, assuming that the source and the target language are aligned perfectly. The second alternative, is to use a neural network which takes as input the current decoder state h_t(note the change in the use of h_t for target sequence hidden state in this case) and predicts a number in the range $[0, T]$ where T is the length of the source sentence. Further the attention weights are also forced to be centered around the value p_t by using a Gaussian distribution whose mean is p_t and standard deviation is $\frac{D}{2}$. Upon training on the WMT 2014 dataset for the English to German translation corpus, the local attention ensemble model was able to provide a 5.0 BLEU point improvement over the non-attentional baseline and on the German-to-English corpus of WMT 2015 dataset, the ensemble model gave a 1.0 BLEU point improvement over the next best NMT system.

Given the discussion we have made so far, it is quite evident that we should be looking towards attention-based models for machine translation, in particular, where the position of the target word is not aligned with the source word position. Attention-based models also provide the required context that is needed for predicting the target word through either local or global attention mechanisms. We expect most(or even all) further research in NMT to have attentional mechanisms with more emphasis on trying to get better at modeling different languages. In the next section, we talk about different representations that can be used for NMT which further pushes the performance of machine translation systems.

6 Beyond Phrase-based Representations in NMT – Dealing with Unknown Words

In prior sections of this paper, we have looked at methods that mostly try to find a good context for generating target words. However, one analysis that has been overlooked so far in this literature review is how to deal with unknown words in both the source and target languages. In the past, unknown words (or out-of-vocabulary words (OOV)) have mostly been dealt with but either generating an *<unk>* symbol for the unknown word or through the use of a back-off to a dictionary. This is clearly not something we would like to encounter when have machine translation systems employed in the real world as it maybe masking a word that carries important information about a sentence. In this section, we look into different methods that have been developed recently to combat this problem of unknown words in different languages encountered during testing.

6.1 Sub-word Representations

As mentioned in the previous paragraph, most of the translation systems that use phrase-based representations use back-off to a dictionary or sometimes even just copy words over from the source language to the target language. This works in the case of names, but suppose our source language contains some compound words, for example the word 'Rajputra' in Hindi can be split as 'Raj—Putra' meaning the king's son in English, it becomes much more easier for the MT system to perform translation when the words are being split up into sub-words.

This has been further investigated in (Sennrich et al., 2016) where the author(s) use the sub-word representation for words in the source language and target language. Through such sub-word representations, the method is also effectively reducing the number of words that would require backing-off to a dictionary and hence pacifies the open-vocabulary issue. To perform the task of word segmentation, the author(s) adapt the Byte Pair Encoding (BPE) algorithm. BPE tries to iteratively replace the most frequent pairs of bytes using a single unused byte. Adapting this to our domain, this would mean that we combine sequence

of frequent characters to develop sub-word units. To construct a vocabulary which the MT system has to learn to use, the vocabulary with only characters available in the training set. Further, words in the training set are represented as a sequence of characters with the '.' symbol at the end to denote end-of-word. Next, the BPE algorithm is used to find the most frequent character pairs in the training set and a merge operation is performed on them to create a new symbol in the vocabulary. In some ways, the construction of this vocabulary using the BPE algorithm is very similar to constructing n-grams. In this case however, we do not consider all the possible n-grams in the dictionary but rather resort to the most frequent ones. Using the word-segmentations constructed from the WMT 2015 dataset for English→German, the author(s) report results on the newstest 2015 dataset for English→German translation. The results indicate that the single model of the sub-word representations can outperform baselines that use back-off to a dictionary and those that produce an <*unk*> symbol for unknown words in terms of BLEU score. With respect to the chRF scores, the ensemble method with sub-word representations are better than other baselines. More importantly the method was able to give comparable or better performance than other methods on rare words and OOV words which essentially proves the effectiveness of using sub-word translations.

6.2 Hybrid Word-Character Representations

In the previous subsection, we have seen how using sub-word representations can help in improving accuracies in OOV word scenarios. One essential component of the approach was to split words into the sub-word representations. We can however take this one step deeper and consult character level embeddings for developing MT systems for the rare words. This was the approach taken by (Luong and Manning, 2016), where the authors use word level embeddings for common words and the character level embeddings for a rare word or an <*unk*> symbol being generated by the translation system as shown in Figure 9. The character-based model is trained using the words available in the training set with a '_' symbol appended at the end to indicate the boundary for the word. The same '_' symbol is also used during generation from the decoder in order to terminate the character generation. During encoding of the source sentence, the rare words are passed through the character model to produce representations. The character models during encoding are initialized with zero hidden state. This is done in order to both maintain simplicity and reduce the training time. During decoding however, when the representation of an <*unk*> symbol is produced by the word level LSTM, the paper employs two approaches to give context to the character level LSTM. The first approach is where the character level LSTM is initialized with the same hidden state as was the word level model which the author(s) refer to as the *same-path* model. In the second approach, the character level LSTM constructs its own context using separate weights for computation which the author(s) also call as the *separate-path* model. For more training details, we would like to refer the reader to (Luong and Manning, 2016). On the English→Czech translation task, the hybrid word-character model was able to outperform all previous baselines with a 20.7 BLEU score. More interestingly, the character-based model alone gave an improvement over the word-based model indicating that character-level translation systems plays a vital role for machine translation in particular for dealing with OOV words. We would like to note that the hybrid model was able to perform better than the character-level model as well. Upon an-

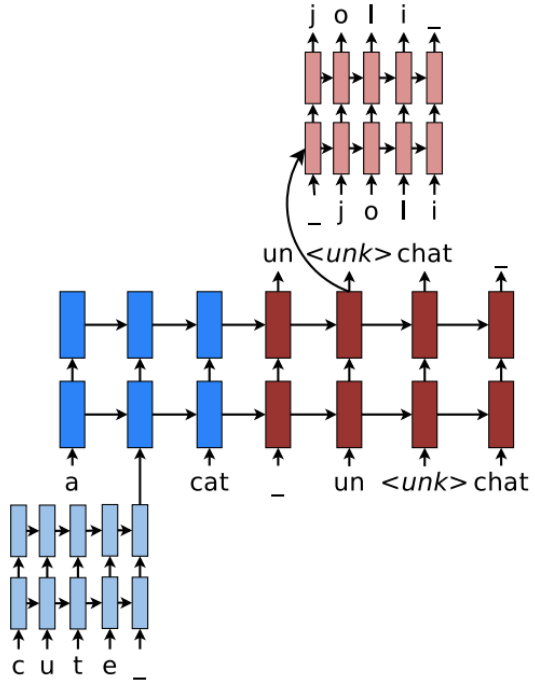

Figure 9: Hybrid Word-Character NMT model. Figure taken from (Luong and Manning, 2016).

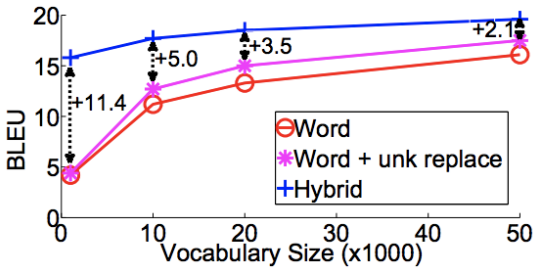

Figure 10: Effect of vocabulary size on translation systems. Figure from (Luong and Manning, 2016).

alyzing the effect of vocabulary size on the ability of word-based models and the hybrid models (see Figure 10), the author(s) were able to find a that when using a lower vocabulary size, the hybrid model provides a BLEU score improvement of about 11.4 BLEU points. Additionally, as the vocabulary size increases there is a small improvement for the hybrid model while still being able to perform better than the word-based models indicating that the hybrid-model is an amazing parameter efficient approach for translation systems. A Barnes-Hut-SNE visualization of words from the *Rare Word Dataset* (see Figure 11), also showed how the hybrid NMT approach is able to get words that have same/similar meanings together. Even words with the same root can be seen to have similar representations.

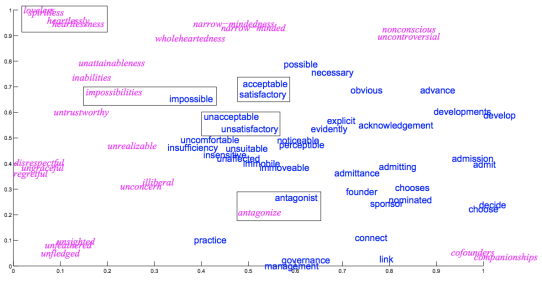

Figure 11: Barnes-Hut-SNE visualization of words. Frequent words are in *blue* and rare words are in *pink*. Figure from (Luong and Manning, 2016).

6.3 Fully Character-based Representations

The results we have seen from the last few subsections are jointly indicative of one fact : we need to use n-gram embeddings in order to tackle the OOV problem. One possible alternative which arises from the hybrid NMT model is to train a fully character level model for machine translation. (Ling et al., 2015) attempted to perform this idea using a bidirectional LSTM, however the results were not so great in comparison to existing models at that time and so many critics shelved the idea.

However, within the previous subsections we have also seen how n-grams have been constructed by either using the BPE algorithm (Sennrich et al., 2016) or by combining word representations with learned unigram(or character-based) representations (Luong and Manning, 2016). Trying to combine the two approaches, we expect that learning n-gram embeddings might perform better for machine translation. This is the motivation for (Lee et al., 2017) where the author(s) propose a model that learns various n-gram embeddings using convolutional neural networks (see Figure 12). Further max-pooling is used to reduce the compress the different n-gram representations to what the author(s) refer to as *segment embedding*. Additional architectural improvement linked to deep learning in particular and training details have can be found in (Lee et al., 2017). For most of the bilingual translation experiments performed in the paper, the proposed method (char2char) gives better BLEU scores than other baselines which use sub-word units (bpe2char). Further more, the multilingual char2char model which converts from multiple source languages to a single target language was able to perform better than some of the bilingual models using both the BLEU score as well as human evaluation using Amazon Mechanical Turk for fluency and adequacy.

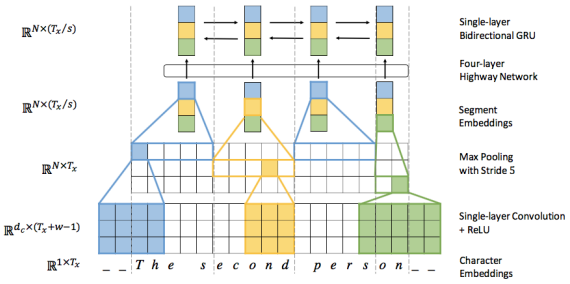

Figure 12: Encoder architecture for model in (Lee et al., 2017).

From the context of OOV words, the paper also gives some qualitative examples where the char2char models seems to give better translations when compared to bpe2char on some rare words and new words situations. These results make the case for using character-level word embeddings

(b) Rare words

DE src	Siebentausendzweihundertvierundfünfzig .
EN ref	Seven thousand two hundred fifty four .
bpe2char	Fifty-five Decline of the Seventy .
char2char	Seven thousand hundred thousand fifties .

(c) Morphology

DE src	Die Zufahrtsstraßen wurden gesperrt , wodurch sich laut CNN lange Rückstaus bildeten .
EN ref	The access roads were blocked off , which , according to CNN , caused long tailbacks .
bpe2char	The access roads were locked , which , according to CNN , was long back .
char2char	The access roads were blocked , which looked long backwards , according to CNN .

Figure 13: Qualitative samples for translations of rare words and new words(morphological changes) by the bpe2char and char2char models (Lee et al., 2017).

for performing translations and even alleviates the problem of OOV word. It also makes a case for learning multi-lingual translation models as the results indicate that such models are less prone to over-fitting to the data.

7 Conclusion

In this paper, we have briefly outlined some of the different approaches that have been adopted to perform machine translation with literature that spans nearly half a century. We would like to reiterate that this review is by no means exhaustive and would like refer the interested reader to (Lopez, 2008; Neubig, 2017) for more in-depth surveys on statistical machine translation. We have also talked about how most of the recent work in machine translation builds upon the existing models from pre-2000. This can be seen in our approach to the neural network models where we constantly make reference the term n-gram models being learned/used in different ways through Section 6. Finally, we have skipped through most of the mathematical equations and numbers(in results) since we would like the paper to be accessible to a wider audience to understand the underlying approaches taken within natural language processing as opposed to "what-works" and the use of large neural networks.

In the future, we expect works in machine translation to use multi lingua training as it has been shown to be effective in reducing overfitting and improving generalization (Lee et al., 2017). Further, building on the idea of making the network learn multiple tasks, we could also use some of the supervised signals proposed in (Linzen et al., 2016) to learn a syntax model for improved translations.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *In the International Conference on Learning Representations, 2015*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguist.*, 16(2):79–85.

Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Mikel L. Forcada, Mireia Ginest-Rosell, Jacob Nordfalk, Jim ORegan, Sergio Ortiz-Rojas, Juan Antonio Prez-Ortiz, Felipe Snchez-Martnez, Gema Ramrez-Snchez, and Francis M. Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

Mikel L Forcada and Ramón P Ñeco. 1997. Recursive hetero-associative memories for translation. In *International Work-Conference on Artificial Neural Networks*, pages 453–462. Springer.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

W. J. Hutchins. 1986. *Machine Translation: Past, Present, Future*. John Wiley & Sons, Inc., New York, NY, USA.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2015. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):8.

Minh-Thang Luong and Christopher D. Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1054–1063, Berlin, Germany. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 133–139, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *CoRR*, abs/1703.01619.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *UNIVERSITY OF MARYLAND, COLLEGE PARK, MD*, pages 20–28.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Jordan B Pollack. 1990. Recursive distributed representations. *Artificial Intelligence*, 46(1-2):77–105.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

Roni Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modeling.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Claude E. Shannon and Warren Weaver. 1949. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana and Chicago.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Warren Weaver. 1949. Warren Weaver Memorandum, July 1949.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto

Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 523–530, Stroudsburg, PA, USA. Association for Computational Linguistics.