# Coarse-to-Fine Attention Models for Document Summarization

Jeffrey Ling

A Senior Thesis for the
Joint Concentration in
Computer Science & Mathematics

Harvard College

March 31, 2017

Dedicated to Leland and Brandon

# Acknowledgements

To Prof. Sasha Rush for his invaluable advising for the past two years. None of this work would be possible without his encouragement and mentorship.

To the graduate students of our lab for a fun and friendly research experience, especially Yoon Kim for his life advice, encyclopedic knowledge of tuning deep neural networks, and constant availability on Slack, and Yuntian Deng for his unyielding optimism that kept my experiments going.

To my thesis readers, Profs. David Parkes and Stuart Shieber, for agreeing to review this incredibly long piece of writing.

To the math library and my cohorts there. Silence is golden.

To my peers for supporting a lively learning environment, especially Rachit for always having a moment for an interesting discussion.

To my roommates, Kenneth and Andres, for not judging me when I returned from the distant land of the quad.

To my parents, who gave their hearts and souls to get me where I am today, and my brothers, who have grown a little bit too tall since I was last home.

To Sindy, for standing by me in just about every way possible.

**Abstract**

While humans are naturally able to produce high-level summaries upon reading paragraphs of text, computers still find such a task enormously difficult. Despite progress over the years, the general problem of document summarization remains mostly unsolved, and even simple models prove to be hard to beat.

Inspired by recent work in deep learning, we apply the sequence-to-sequence model with attention to the summarization problem. While sequence-to-sequence models are successful in a variety of natural language processing tasks, the computation does not scale well to problems with long sequences such as documents. To address this, we propose a novel coarse-to-fine attention model to reduce the computational complexity of the standard attention model.

We experiment with our model on the CNN/Dailymail document summarization dataset. We find that while coarse-to-fine attention models lag behind state-of-the-art baselines, our method learns the desired behavior of attending to subsets of the document for generation. Therefore, we are optimistic that the general approach is viable as an approximation to state-of-the-art models. We believe that our method can be applied to a broad variety of NLP tasks to reduce the cost of training expensive deep models.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Natural Language Processing

The field of natural language processing arises from a very simple question: how can we teach machines to read, speak, and understand the words that we use with such ease and fluency?

Such a question has been considered since the first computers were built. The classic Turing test, posed by Alan Turing in 1950, requires a machine to converse in a way that is indistinguishable from a human, and thus requires a fundamental grasp on how to properly use language. Although it was simple for Turing to conceptualize what a successful machine might look like, many have been stumped on how to actually construct such a system. Indeed, to this day, no machine has been able to fully pass the Turing test as it was originally posed.

While computers can now run computations at a rate that far exceeds human cognition, language tasks that we consider trivial still prove to be extraordinarily difficult for a machine to solve. Consider the problem of deciding words with multiple meanings such as "bass" (word sense disambiguation), or the problem of identifying to what or whom a certain pronoun refers (coreference). While humans reliably perform these functions on a daily basis, they are not at all easy for computers to handle.

However, the need for computers to understand language has never been greater. In today's information age, NLP grows increasingly important as the accumulation of free-form text begins to outpace the ability of humans to process it. In fields such as medicine, this can mean missed diagnoses; in law, wasted effort on irrelevant documents; in international relations, misinterpretations of foreign articles. Because natural language

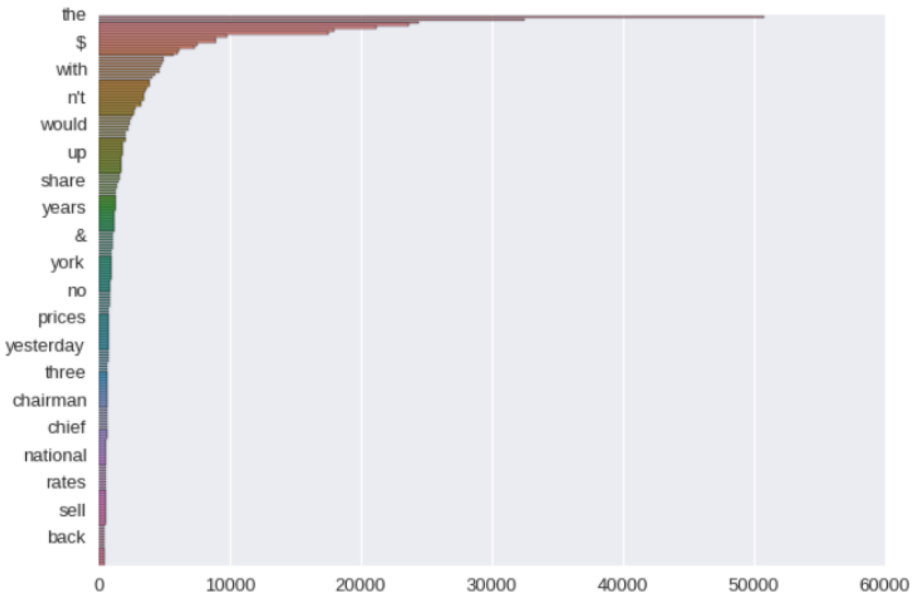



*Figure 1.1: Zipf's law, indicating the relationship between English words and their frequency. The distribution approximately follows an inverse power law. Image from CS287.*

is everywhere and used by everyone, the demand for text processing solutions remains as high as ever.

## 1.2 Methods in NLP

In this section, we investigate the general philosophy of tackling NLP problems.

The key property of language that makes it difficult for machines to handle is its discrete and combinatorial nature. When we form sentences, we can string together arbitrary words from our vocabulary, as long as we follow the rules of some highly structured grammar. In some sense, the essential difficulty in NLP lies in handling these complicated structures in an efficient way.

Linguistics attempts to provide an answer by building a formal theory of language. Indeed, many ideas from linguistics, including sentence parses, morphology, and semantics are invaluable in NLP for understanding how sentences and phrases are put together. In order to solve a language problem, we might begin by enriching the surface form of text (the raw sentences and paragraphs) with the parse structure, parts of speech, coreferent entities, etc. We can then use this featurized form of language in whichever way we prefer.

Another mode of thought focuses on the role of statistics in language. A cursory examination of the distribution of English words reveals an interesting power law distribution known as Zipf's law (Figure 1.1). By drawing from ideas of information theory (Shannon, 1948), we can treat language as the result of some noisy probabilistic process, and we can reduce problems such as language modeling to learning the parameters of some simple distributions.

Historically, there has been contention about the roles of linguistics and statistics in NLP. Certain practical problems seem to be better off without linguistic theory; as IBM researcher Frederick Jelinek famously said, "Every time I fire a linguist, the performance of the speech recognizer goes up" (Jurafsky & Martin, 2009). Indeed, even naive models of language perform well if given enough training data, as evidenced by the IBM models for machine translation (Brown et al., 1993).

Today, the linguistics-free approach has been taken to an extreme by deep learning systems. For the first time, neural networks are able to learn to perform language tasks in an end-to-end fashion (Collobert et al., 2011b), i.e. without the linguistic preprocessing that was once considered necessary. Neural methods have been adopted by user-facing systems like Google Translate with great success (Wu et al., 2016).

## 1.3 Automatic Summarization

In this thesis, we will consider the particular problem of *text summarization*. That is, given a document with several sentences of text, the goal is to produce a concise and fluent summary that captures most of its salient points.

Text summarization is one of the biggest open problems in NLP, as identifying the key information of a document seems to require a deep understanding of its content. Nevertheless, there is an urgent need for successful summarization algorithms in the text-dominated discourse of the modern era.

Nenkova & McKeown (2011) give a comprehensive overview of the problem of summarization. In particular, they provide a taxonomy of methods that researchers have developed to tackle the task:

**Extractive vs. abstractive** Extractive summaries extract certain sentences or phrases from the document with minimal further processing, while abstractive summaries might

include paraphrasing and original word choice (similar to how humans produce summaries).

Extractive methods are by far more popular due to the simplicity of building an extraction algorithm. Abstractive methods are much more difficult, as we need to both capture the critical pieces of information and also ensure grammaticality of the output (a difficult problem in itself).

**Single- vs. multi-document** Summarization was originally posed as the problem of producing a summary for a single document. However, with the onset of the Internet, there are often multiple documents on the same topic, and so a summarization system should be able to combine them to produce a summary.

Interestingly, Nenkova & McKeown (2011) note that the multi-document problem is often easier due to redundant information across sources. In contrast, in the single-document problem, a crucial piece of knowledge might only be repeated once.

**Generic vs. query-focused** Generic summaries make no assumptions about the reader and are meant to be generally informative, while query-focused summaries take into consideration a query and only return relevant information from the document.

The contrast between these two methods highlights an important question in summarization: to what end are we summarizing documents? If we can answer this question, we can build systems to more accurately accomplish our desired tasks.

Most of this taxonomy is based upon insights from DUC (Document Understanding Conferences), a set of tasks released by NIST between 2001-2006 to promote research in summarization (Over et al., 2007).

The DUC tasks were fairly diverse and varied from year to year. DUC 2001 and 2002 asked for generic summaries of news articles, while DUC 2003 presented a multi-document problem. DUC 2004-2006 shifted to a more question-answering based approach, and many of the documents were accompanied by focused queries.

While DUC did not lead to any definitive answers on how best to summarize documents, some important empirical discoveries were noted. For the generic summarization tasks, it was observed that the first sentence of news articles was a strong baseline that more sophisticated methods found hard to beat. Thus, the generic summary task was seen as ill-formed and not as interesting, leading to the more query-focused tasks in the later years.

DUC also inspired a lot of thinking on how to best evaluate summaries. Evaluating a summary is inherently ambiguous, as even humans tend to disagree on what makes for a good summary. While a single best quantitative metric may not exist, DUC established several important criteria, including grammaticality, non-redundancy, and content coverage.

One proposed metric for evaluation is recall on elementary discourse units (EDUs), labeled clauses within a summary that ought to be captured. Unfortunately, this method requires the data to be labeled with clauses. ROUGE (Lin, 2004), inspired by the BLEU metric for machine translation (Papineni et al., 2002), is a cheap and fast method based on overlapping n-grams, and does not require any additional annotation. Due to its ease of use, ROUGE is one of the most popular evaluation metrics today.

While useful, none of these metrics directly address the grammaticality of the output. Aside from using human evaluation, meaningful metrics for summaries is still very much an open question (Toutanova et al., 2016).

Going forward, we will limit our scope to the single-document and generic summarization case. We will explore some related work in extractive and abstractive summarization, then connect these approaches with recent trends in deep learning.

## 1.4 Related Work

One of the first treatises on automatically producing summaries was Luhn (1958), which considered the problem of producing abstracts for scientific articles. At the time, computers were still monolithic machines that ran on punch cards, so automating the summarization process was quite ahead of its time.

Luhn (1958) proposed a simple sentence-ranking method to produce summaries. The algorithm gives each sentence a score based on the occurrence of frequently appearing words. This is one of the first examples of an extractive summarization method.

Since then, a variety of approaches have been applied. We highlight some notable work in both the extractive and abstractive frameworks.

### 1.4.1 Extractive

The most popular methods for document summarization have generally been extractive due to their simplicity.

There are two natural procedures for extractive summarization: one is to produce a ranking of sentences based on some scoring function, and the other is to train a binary classifier on whether a sentence should be in the summary. Luhn (1958) is an example of the first approach.

Carbonell & Goldstein (1998) extend on the scoring-ranking method with an information metric, MMR (maximum marginal relevance), that penalizes pairwise similarity between sentences. Their work is one of the first that attempt to reduce redundancy in the summary.

Kupiec et al. (1995) pioneer the second method by treating sentence selection as a classification problem, training a naive Bayes classifier on sentences.

Ranking and classifying algorithms have generally become more sophisticated over time. Today, deep learning systems prove to be some of the most powerful classifiers, and some have found applications in extractive summarization. Cao et al. (2015) use convolutional neural networks to extract features for each sentence, combining these with document-level features to produce ranking scores. Cheng & Lapata (2016) apply the encoder-decoder model using recurrent neural networks to jointly produce labels for each sentence.

One key drawback of these trained extractive models, especially the deep ones, is the scarcity of annotated data. In extractive models, we require sentences of each document to be labeled as positive or negative in order to train a classifier, and data in this format can be hard to obtain.

### 1.4.2 Abstractive

While extraction has proven to be successful, the method is inherently limited in its ability to summarize. The more challenging method, and also the closest to what humans do, is *abstractive* summarization. Instead of strictly requiring the summary to be a subset of the source document, any coherent text is allowed.

Because generative models of language were historically not very powerful, the main techniques used to produce abstractive summaries involved extraction followed by sentence compression. Knight & Marcu (2002) employ a noisy channel model and probabilistic context-free grammars to deduce the "most probable" compression of a sentence. Cohn & Lapata (2008) extend the grammar-based method to allow for insertions and substitutions during compression, whereas prior methods were purely deletion based. Zajic et al. (2004) combine sentence compression with an unsupervised topic detection algo-

rithm to achieve the best result on DUC 2004. Recently, Durrett et al. (2016) apply an integer linear programming (ILP) approach, where they extract phrases according to a feature-based scoring function and merge them with grammaticality constraints.

We note that because these methods rely heavily on a strong extraction system, they might not qualify as truly "abstractive." Deep learning models, however, have been very successful at building generative models of language, and have naturally found applications in abstractive summarization. Rush et al. (2015) propose a completely data-driven model for headline generation by training an end-to-end model. More recently, Nallapati et al. (2016) apply the same approach on full documents.

As with deep learning methods for extraction, these models require a large amount of supervised training data. One advantage of the abstractive model is that data in the document-abstract format is more easily obtainable than labeled extractive data.

### 1.4.3 In the Wild

Outside of the academic realm, summarization is an important problem in industry. One noteworthy summarization method is on Reddit[1]: in order to summarize long forum discussions, Reddit uses technology from Smmry[2].

Smmry's algorithm is a simple extractive method. It counts word occurrences, splits discussions by sentence, and ranks the sentences based on the sum of their word scores. This algorithm bears extraordinary similarity to Luhn (1958) — although a variety of work has been done since then, the simplest approaches turn out to be the most practical.

## 1.5 This Work

Inspired by advances in deep learning for NLP and dissatisfied by the limitations of extractive models, we set out to build a deep model to abstractively generate summaries for documents. Because deep learning is computationally expensive even for short lengths of text, this creates a challenge for the general document summarization problem.

Hence, we will survey the literature for methods that alleviate the computation of training deep models. Along the way, we propose a new model architecture that extends the popular sequence-to-sequence attention model in NLP and attempts to reduce the computational complexity of the document summarization problem.

[1] reddit.com
[2] smmry.com

## 1.6 Outline

We provide an outline for the rest of this thesis.

In Chapter 2, we give a survey of deep learning and motivate its use in solving our problem. We also provide the necessary background material for understanding our models and algorithms. In Chapter 3, we describe our training algorithm formally. In Chapter 4, we describe our models formally. In Chapter 5, we describe the experimental setup, including our dataset and baselines. In Chapter 6, we show results and analyze the outputs of our models. In Chapter 7, we discuss our results. Finally, we conclude in Chapter 8.

# Chapter 2

# Background

In this chapter, we give a brief primer on deep learning and representation learning in the context of NLP.

Then, we set up the relevant background ideas for our models. We describe the popular sequence-to-sequence attention model at a high level, then survey the literature for methods in reducing the computational cost of deep models. Finally, we introduce the framework of reinforcement learning, which will provide the foundation for one of these methods.

## 2.1 Deep Learning

The history of neural networks dates back to the perceptron (Rosenblatt, 1958), a simple model that assumes data can be linearly separated. Due to this strict requirement, the machine learning community dismissed the idea as impractical for most of the 20th century.

Recently, neural networks have made a resurgence. In the ImageNet image classification competition in 2012, Krizhevsky et al. (2012) won using deep convolutional neural networks (LeCun & Bengio, 1995), beating the competition by a significant margin. This led to a renewed wave of research, especially due to the advancement of GPU computing power, which can train networks at 10 to 20 times the speed of standard CPUs. Today, deep models are successfully used in image recognition (Farabet et al., 2013), speech recognition (Hinton et al., 2012), and Go playing (Silver et al., 2016), just to name a few.

In NLP, Bengio et al. (2003) first demonstrated the viability of deep models by building a neural language model using multi-layered perceptrons. Later, Collobert et al. (2011b)

show that neural models can be used to train end-to-end models without any of the preprocessing that was once considered necessary.

Since their onset, deep models have found their way into nearly every corner of NLP. Much of their success relies on the ubiquity of the *long short-term memory* (LSTM) recurrent neural network (Hochreiter & Schmidhuber, 1997), a model used to both process and generate sequences of text, and *word vectors*, distributed vector representations of text. Several state-of-the-art algorithms are now based upon these deep learning tools; while there is too much to cover here, Goldberg (2015) provides a concise summary of the models that have had the greatest impact on NLP.

## 2.2 Representation Learning

While neural networks are often treated as black box classifiers, Zeiler & Fergus (2014) show that the intermediate layers of deep convolutional networks in computer vision contain abstracted qualities of the input, such as patterns, textures, and objects. This suggests that neural networks are discovering latent features and building generalized *representations* of their inputs.

The idea of learning representations of the input also applies in NLP. Mikolov & Dean (2013) show that by training an unsupervised neural network on a Google News text corpus, the network learns to map words in the English language to vectors of real numbers known as *word embeddings*. These word embeddings are actually able to capture semantic properties of the words — for example, taking the vectors for *king*, *man*, and *woman*, we find that $v_{king} - v_{man} + v_{woman} \approx v_{queen}$, preserving the analogy that we usually make in English. Figure 2.1 shows some of the word2vec vectors when projected to two dimensional space.

Representation learning has since become a central topic in NLP. Word vectors trained on a general text corpus can be used in almost any NLP model (Mikolov & Dean, 2013; Pennington et al., 2014), and the task-invariant property of these vectors is highly attractive. It remains to be seen whether it's possible to represent longer pieces of text, such as sentences, in a general way, and research in the area is active (Bowman et al., 2016).

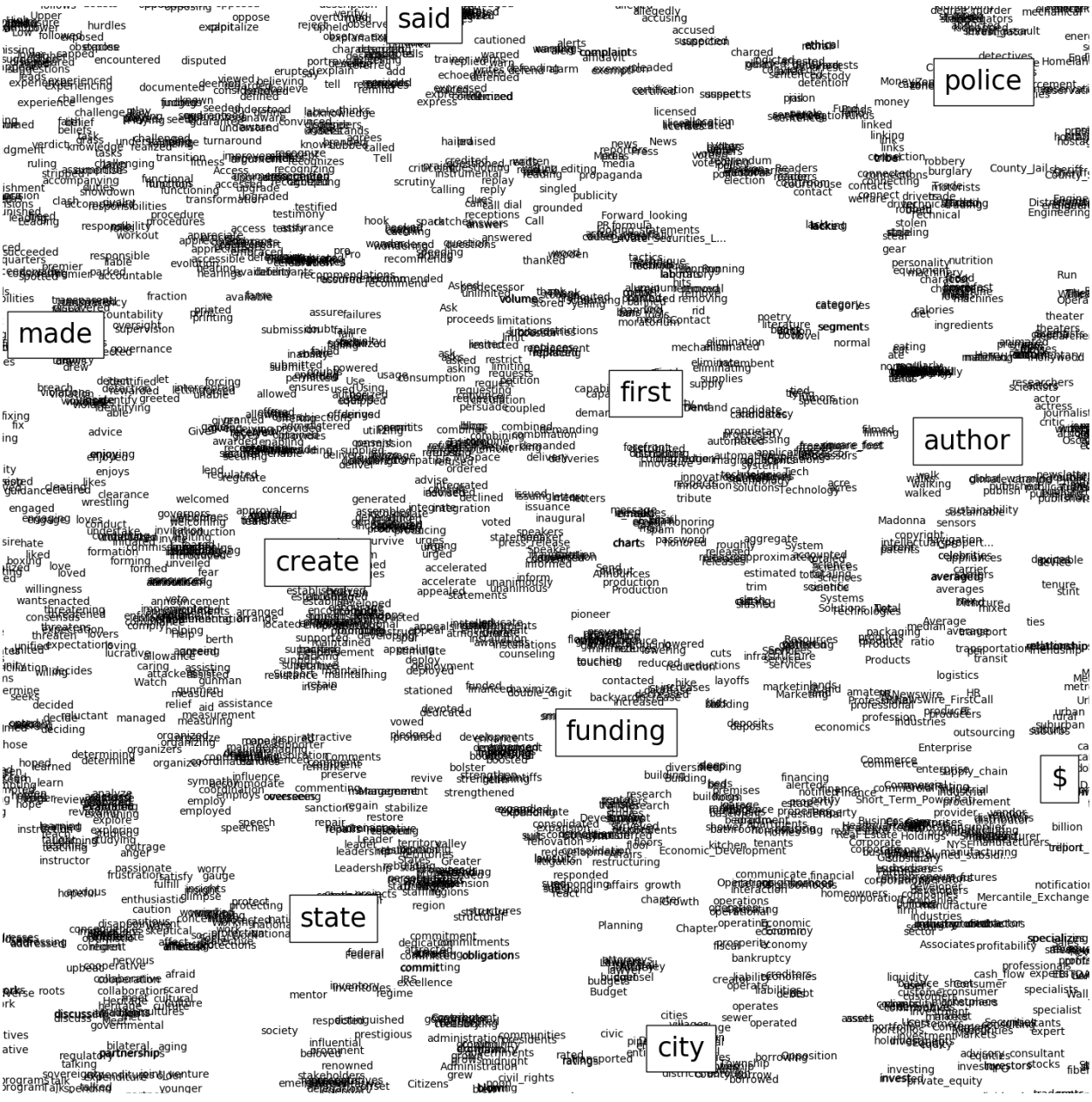



*Figure 2.1: A visualization of word2vec word vectors (Mikolov & Dean, 2013), projected to two dimensional space using PCA. Words with similar semantic meaning tend to cluster together in the space. Image from CS287.*

## 2.3 Motivation

Why deep models? There are currently two general approaches to solving problems in NLP: one is to use as much linguistic theory as possible to reduce the problem, and another is to apply black box learners such as deep neural networks.

Deep models work fantastically well on many tasks, especially when it comes to improving on metrics, and one might wonder if there is even a need to consider other models. However, as big of a hammer as deep learning is, it is useless without nails to hit. That is, in order to further language understanding, datasets and tasks must be posed such that deep models can be applied; it is exactly in this domain that classical theory is still relevant. As Manning (2016) argues, although neural models have come to dominate NLP papers, there will always be a need for domain experts to prepare the field so that deep learning can succeed.

With this caveat, there are many worthwhile reasons to study deep networks in NLP. First, they work! In fact, they work remarkably well without any feature engineering, which tends to be one of the fussiest parts of building machine learning algorithms.

Second, they are not mutually exclusive with standard feature extraction methods, and so can augment classical methods.

Third, we find that trained models can discover latent structure in language automatically, which may reveal insights about how language is used. For example, word vectors tend to cluster around semantic concepts without any direct supervision.

It is this third point upon which we base this thesis. Although they began as black-box optimizers, end-to-end deep models are slowly being dissected into more understandable parts. One of our goals is to test the hypothesis that such parts are in fact interpretable and are functioning as we expect them to.

With this goal in mind, we attempt to interpretably extend the attention mechanism of the popular *sequence-to-sequence models*. In the next section, we describe the sequence-to-sequence model informally.

## 2.4 Sequence-to-Sequence Attention Models

Many NLP problems can be posed as follows: given an input sequence of tokens $\mathbf{x} = x_1, \ldots, x_n$ with $x_i \in \mathcal{X}$, we train a model to produce an output sequence $\mathbf{y} = y_1, \ldots, y_m$ with $y_j \in \mathcal{Y}$. We normally pose this as a probabilistic problem and model the conditional

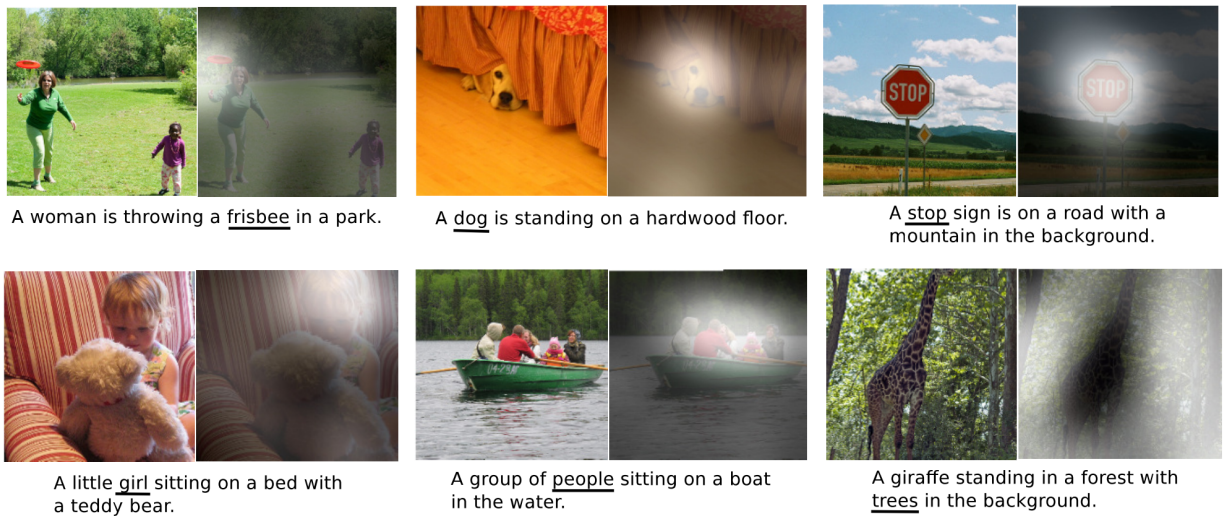



*Figure 2.2: Attention of a caption generating neural model (Xu et al., 2015). The model learns to highlight the correct object when generating words.*

probabilities, so that we wish to find

$$\begin{aligned} &\arg\max_{\mathbf{y}\in\mathcal{Y}^m} p(y_1,\ldots,y_m|x_1,\ldots,x_n) \\ &= \arg\max_{y_1,\ldots,y_m\in\mathcal{Y}} p(y_1|\mathbf{x})p(y_2|y_1,\mathbf{x})\cdots p(y_m|y_1,\ldots,y_{m-1},\mathbf{x}) \end{aligned} \tag{2.1}$$

The sequence-to-sequence architecture of Sutskever et al. (2014), also known as the encoder-decoder architecture or seq2seq, neatly provides a solution to this framework. By encoding the input $x_1,\ldots,x_n$ into a fixed size vector which we call the *context vector*, we can compute the conditional probabilities and hence generate $y_1,\ldots,y_m$. This model has been used to great effect in a variety of NLP tasks, including machine translation (Sutskever et al., 2014; Bahdanau et al., 2014), question answering (Hermann et al., 2015), dialogue (Li et al., 2016), caption generation (Xu et al., 2015), and in particular summarization (Rush et al., 2015).

A popular variant of sequence-to-sequence models are *attention* models (Bahdanau et al., 2014). Instead of mashing the input into a single context vector, we instead keep an encoded representation of each part of the input, "attending" to the relevant part each time we produce an output from the decoder. In practice, this means computing attention weights for all encoder hidden states, then taking the weighted average as our new context vector.

Xu et al. (2015) show how attention models can be used to "summarize" an image and produce a caption (Figure 2.2). By analyzing where in the image their models attend to when generating each word of the caption, i.e. where the attention weights are highest in the image, they qualitatively find that the model is essentially describing certain objects.

While successful, existing seq2seq methods are limited by the length of source and target sequences. For a problem such as document summarization, the source sequence of length $N$ requires $O(N)$ model computations to encode, where $N$ could potentially be very large. However, it makes sense intuitively that not every word of the document will be necessary for generating a summary, and so we would like to reduce the amount of necessary computations over the source.

Therefore, in order to scale seq2seq methods for this problem, we aim to prune down the length of the source sequence in an intelligent way. The natural solution is to force the model to only use a subset of the input rather than naively encoding the entire input. We investigate some related work in this area.

## 2.5 Conditional Computation

Many techniques have been proposed in the literature to efficiently handle the problem of large inputs to deep neural networks, and we consider the particular framework of conditional computation.

The term "conditional computation" was coined by Bengio et al. (2013) — the idea is to only compute a subset of a network for a given input. To this end, Bengio et al. (2013) propose the use of stochastic nodes in the network to implement discrete random variables, which serve as gates to turn on or off certain parts of the network for computation. Unfortunately, discrete variables cannot be backpropagated through as they produce zero gradient, and so Bengio et al. (2013) suggest the naive *straight-through* estimator for binary stochastic gates. In the forward step of the network, they simply sample from the stochastic gates, and backpropagate the gradient as if they had not sampled.

Many alternative methods, some deterministic and some stochastic, have been proposed for conditional computation.

On the stochastic front, we usually want to sample from a multinomial distribution, i.e. select one choice out of multiple choices, to conditionally compute. In deep networks, the softmax function $\text{softmax} : \mathbb{R}^m \to [0,1]^m$ is used to produce a normalized probability

distribution out of any vector of reals. For $\mathbf{z} \in \mathbb{R}^m$,

$$\text{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{2.2}$$

We can sample from this distribution to decide which parts of the network to compute. To train the model, we will require methods from reinforcement learning, which we cover in the next section. Xu et al. (2015) apply this technique in the form of "hard" attention — while standard "soft" attention averages the representations of where the model attends to, in hard attention we discretely select a single location. This approach has also been applied by others for computer vision tasks (Mnih et al., 2014; Ba et al., 2015).

On the deterministic front, Rae et al. (2016) use an approximate nearest neighbors approach for their "sparse access memory" model to train a large-scale neural Turing machine. Shazeer et al. (2017) introduce a mixture-of-experts model that deterministically chooses a subset of "expert" networks to train at any given time. The key-value memory networks of Miller et al. (2016) select subsets of text by computing word overlap; this process, however, is not learned. Martins & Astudillo (2016) propose the *sparsemax* function as a substitute for softmax, which projects a given vector of weights to the probability simplex. It turns out that this function is differentiable and also has sparse output; however, we are not guaranteed to have a one-hot vector as we get from sampling the multinomial distribution.

## 2.6 Reinforcement Learning

Standard backpropagation training of neural networks assumes that the output is a deterministic and differentiable function of the input. Reinforcement learning, however, is a more general framework that makes no such assumptions.

The traditional setup of reinforcement learning (RL) assumes some agent is navigating an environment and earning rewards. We assume a state space $\mathcal{S}$, an action space $\mathcal{A}$, a reward function of state and action $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a Markovian transition distribution $p(s'|s, a)$ for $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$.

We suppose that at time $t$, the agent is in state $s_t \in \mathcal{S}$, makes an action $a_t \in \mathcal{A}$, earns a reward $r_t = R(s_t, a_t)$, and transitions probabilistically to the next state $s_{t+1}$. We suppose the agent has a *policy function* $\pi : \mathcal{S} \to \mathcal{A}$ that it uses to make decisions. Then the agent

wants to maximize total expected reward

$$\mathbb{E}_{s_t,a_t}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$$

by finding the optimal policy $\pi$. Here, $\gamma \in (0,1)$ is a time discount factor for the reward, and we assume an infinite time horizon.

In general RL, we assume that we don't know the reward function $R(s,a)$ or the transition distribution $p(s_{t+1}|s_t,a_t)$. While the environment still gives us rewards for actions, finding the optimal policy requires predicting which states lead to those rewards. Since we don't know ahead of time which states are best, we must explore the state space to find what actions lead to the best rewards.

There are several methods for solving the general RL problem. We consider the approach of directly learning the policy function $\pi(\cdot;\theta) : \mathcal{S} \to \mathcal{A}$, where $\pi$ is now parametrized by weights $\theta$. We can train $\pi$ to maximize expected reward through a gradient ascent method known as *policy gradient*, or the REINFORCE algorithm (Williams, 1992).

REINFORCE can be used to train deep neural networks with stochastic units by a simple extension of the backpropagation algorithm. In the next chapter, we derive the generalized training algorithm and connect it to our models.

# Chapter 3

# Algorithms

While reinforcement learning is an attractive framework for posing our models, the details of training are slightly complicated in the context of deep learning. In this chapter, we describe how both deep learning and reinforcement learning both fit into the rigorous model of stochastic computation graphs. We then give the corresponding training algorithm in detail.

## 3.1 Stochastic Computation Graphs

Neural networks with stochastic units are also known as *stochastic computation graphs*, as defined by Schulman et al. (2015). Formally, they define a stochastic computation graph (SCG) as a directed, acylic graph with three kinds of nodes:

1. Input nodes, including fixed network inputs and parameters.
2. Deterministic nodes, which are deterministic functions of their parents.
3. Stochastic nodes, which are random variables distributed conditionally on their parents.

We can formulate a training problem for SCGs by choosing certain terminal nodes and taking their sum as the objective function. If a node is stochastic, we take its expectation. That is, for a set of terminal nodes $\mathcal{C}$, we optimize $\sum_{c \in \mathcal{C}} \mathbb{E}[\hat{c}]$, where $\hat{c}$ denotes the random variable corresponding to node $c$.

Note that the SCG formulation captures both supervised deep learning and RL policy functions. Deep neural networks are just SCGs with all deterministic nodes, and the loss

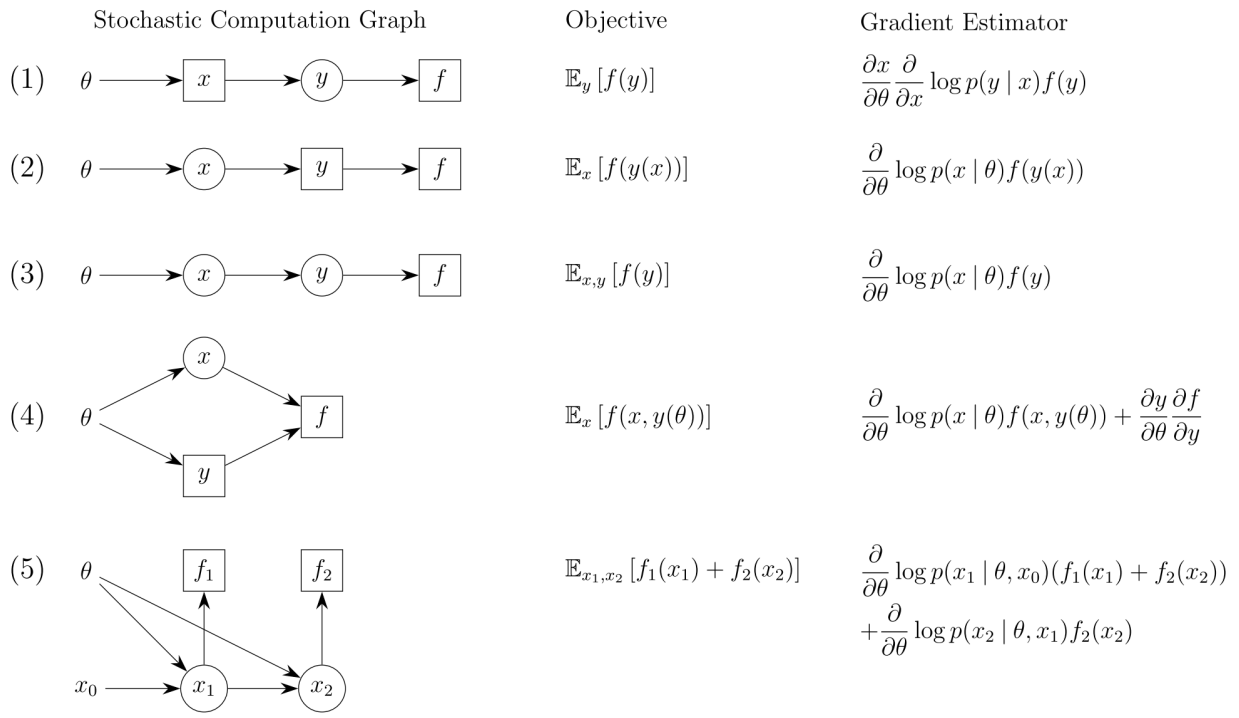



*Figure 3.1: Simple examples of stochastic computation graphs (SCGs), along with the objective and corresponding gradient estimator. Nodes in circles are stochastic, nodes in squares are deterministic, and the rest are input nodes. Diagrams from Schulman et al. (2015).*

function $\mathcal{L}$ of the task is the objective. The RL policy function $\pi$ is an SCG with actions at the stochastic nodes, and the expected reward $\mathbb{E}[r]$ is the objective.

Figure 3.1 shows examples of simple SCGs, along with their corresponding objectives. To optimize the objective, we want to perform gradient descent, and so for any parameter $\theta$, we need to derive an estimator of its gradient.

We see in these special cases that there is an intuitive pattern to the gradient estimators. If there is a path from $\theta$ to the objective that goes through a stochastic node, we get a term for the derivative of the log probability multiplied by the objective. For paths through deterministic nodes, we get standard derivatives as in the chain rule from backpropagation.

Next, we derive the gradient of the objective with respect to parameter input nodes in the general case.

## 3.2 Deriving the Gradients

We define for nodes $w, v$ the relation $w \prec v$ (pronounced "$w$ influences $v$") if there exists a path from $w$ to $v$ in the SCG. We also say $w \prec^D v$ ("$w$ deterministically influences $v$") if there exists a path that consists of only deterministic nodes.

Given an SCG, let $\mathcal{S}$ be the set of stochastic nodes and $\mathcal{C}$ the set of objective or cost nodes. For a given node $v$, let $\text{par}(v)$ denote its parents.

For a given parameter $\theta$, we have the following:

**Theorem 1.** *Assume differentiability of all functions. Then the gradient of the objective with respect to $\theta$ is*

$$\frac{\partial}{\partial\theta}\mathbb{E}\left[\sum_{c\in\mathcal{C}}\hat{c}\right] = \mathbb{E}\left[\sum_{c\in\mathcal{C}}\hat{c}\sum_{v\in\mathcal{S},\ \theta\prec^D v,\ v\prec c}\frac{\partial}{\partial\theta}\log p(v|\,\text{par}(v);\theta) + \sum_{c\in\mathcal{C},\ \theta\prec^D c}\frac{\partial}{\partial\theta}\hat{c}\right] \tag{3.1}$$

$$= \mathbb{E}\left[\sum_{v\in\mathcal{S},\ \theta\prec^D v}\left(\frac{\partial}{\partial\theta}\log p(v|\,\text{par}(v);\theta)\right)\hat{C}_v + \sum_{c\in\mathcal{C},\ \theta\prec^D c}\frac{\partial}{\partial\theta}\hat{c}\right] \tag{3.2}$$

*where $\hat{C}_v = \sum_{c\in\mathcal{C},\ v\prec c}\hat{c}$ is a random variable of the cost that stochastic node $v$ influences.*

Before we give the proof, we note the intuition. The first term represents the gradient from the stochastic nodes, where we multiply the gradient of the log probability with the cost. The second term is the standard gradient we obtain from backpropagation, summed over all cost nodes that $\theta$ deterministically influences.

*Proof.* Due to linearity of expectation, we only need consider a single node $c \in \mathcal{C}$. Let $\mathcal{L} = \mathbb{E}[\hat{c}]$ be the loss function.

We have stochastic nodes $v$ such that $\theta \prec^D v$ and $v \prec c$, i.e. there is a path from $\theta$ to $c$ through $v$. Let this set be $\mathcal{S}_\theta$, and denote the joint random variable of this set as $\hat{\mathbf{v}}$. Let $c(\hat{\mathbf{v}};\theta)$ explicitly denote $\hat{c}$.

We compute the gradient:

$$\begin{aligned}
\frac{\partial}{\partial\theta}\mathbb{E}[\hat{c}] &= \frac{\partial}{\partial\theta}\sum_{\hat{\mathbf{v}}} p(\hat{\mathbf{v}}|\,\text{par}(\mathbf{v});\theta)\cdot c(\hat{\mathbf{v}};\theta) \\
&= \sum_{\hat{\mathbf{v}}}\frac{\partial p(\hat{\mathbf{v}}|\,\text{par}(\mathbf{v});\theta)}{\partial\theta}\cdot c(\hat{\mathbf{v}};\theta) + p(\hat{\mathbf{v}}|\,\text{par}(\mathbf{v});\theta)\cdot\frac{\partial}{\partial\theta}c(\hat{\mathbf{v}};\theta) \\
&= \sum_{\hat{\mathbf{v}}}\frac{\partial p(\hat{\mathbf{v}}|\,\text{par}(\mathbf{v});\theta)}{\partial\theta}\cdot c(\hat{\mathbf{v}};\theta) + \mathbb{E}_{\mathbf{v}}\left[\frac{\partial}{\partial\theta}c(\hat{\mathbf{v}};\theta)\right]
\end{aligned}$$

Note that the second term gives the standard backpropagation gradient. We can rewrite the first term:

$$\begin{aligned}
\sum_{\hat{\mathbf{v}}} \frac{\partial p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)}{\partial\theta} \cdot c(\hat{\mathbf{v}};\theta) &= \sum_{\hat{\mathbf{v}}} p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta) \frac{1}{p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)} \frac{\partial p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)}{\partial\theta} \cdot c(\hat{\mathbf{v}};\theta) \\
&= \sum_{\hat{\mathbf{v}}} p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta) \frac{\partial \log p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)}{\partial\theta} \cdot c(\hat{\mathbf{v}};\theta) \\
&= \mathbb{E}_{\hat{\mathbf{v}}\sim p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)} \left[ \frac{\partial \log p(\hat{\mathbf{v}}|\operatorname{par}(\mathbf{v});\theta)}{\partial\theta} \cdot c(\hat{\mathbf{v}};\theta) \right] \\
&= \mathbb{E}\left[ \widehat{c} \cdot \sum_{v\in\mathcal{S}_\theta} \frac{\partial \log p(\hat{v}|\operatorname{par}(v);\theta)}{\partial\theta} \right]
\end{aligned}$$

where the last equality follows from separating the log joint probability term into its conditional marginals.

We have thus shown Equation 3.1. Equation 3.2 follows directly by rearranging the order of summations, noting which nodes $v$ influence a certain $c$.

□

Because the expectations in both gradient formulae are intractable in the general case, we use a single Monte Carlo sample as our gradient estimator. This estimator is unbiased; however, the variance of the Monte Carlo gradient can be very high in practice.

One of the simplest ways to reduce the variance of the gradient estimator is to introduce a baseline scalar $b \approx \mathbb{E}[\hat{c}]$ which we subtract from each cost term. That is, we have:

**Theorem 2.** *The gradient of Equation 3.1 from Theorem 1 can also be written as*

$$\frac{\partial}{\partial\theta}\mathbb{E}\left[\sum_{c\in\mathcal{C}} \hat{c}\right] = \mathbb{E}\left[\sum_{c\in\mathcal{C}} (\hat{c} - b_c) \sum_{v\in\mathcal{S},\ \theta\prec^D v,\ v\prec c} \frac{\partial}{\partial\theta} \log p(v|\operatorname{par}(v);\theta) + \sum_{c\in\mathcal{C},\ \theta\prec^D c} \frac{\partial}{\partial\theta}\hat{c}\right] \tag{3.3}$$

*where $b_c$ is a scalar not influenced by any of the $v$ in the summation.*

*Proof.* Equation 3.3 is identical to Equation 3.1, except for the added terms

$$\mathbb{E}\left[ b_c \sum_{v\in\mathcal{S},\theta\prec^D v, v\prec c} \frac{\partial}{\partial\theta} \log p(v|\operatorname{par}(v);\theta) \right]$$

Considering a single $c$, by linearity it suffices to show that

$$\mathbb{E}\left[\frac{\partial}{\partial\theta}\log p(v|\operatorname{par}(v);\theta)\right] = 0, \qquad \forall v \in \mathcal{S}_\theta$$

This is true since the $b_c$ term is constant with respect to the expectation (by our assumption that no $v$ influences it).

But this follows since

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial}{\partial\theta}\log p(\hat{v}|\operatorname{par}(v);\theta)\right] &= \sum_{\hat{v}} p(\hat{v}|\operatorname{par}(v);\theta)\frac{\partial \log p(\hat{v}|\operatorname{par}(v);\theta)}{\partial\theta} \\
&= \sum_{\hat{v}} p(\hat{v}|\operatorname{par}(v);\theta)\frac{1}{p(\hat{v}|\operatorname{par}(v);\theta)}\frac{\partial p(\hat{v}|\operatorname{par}(v);\theta)}{\partial\theta} \\
&= \frac{\partial}{\partial\theta}\sum_{\hat{v}} p(\hat{v}|\operatorname{par}(v);\theta) = \frac{\partial}{\partial\theta}[1] = 0
\end{aligned}$$

Thus, the gradient with baseline is unbiased. □

Including a baseline is proven to reduce the variance of the estimator (Weaver & Tao, 2001). There are several different methods for producing baselines, such as taking the average of all previously seen cost terms, and we will not cover all of them here.

## 3.3 Training

In order to compute the gradients of SCGs in practice, we make a slight tweak to the backpropagation algorithm.

We first perform a forward pass of the SCG to obtain all values and samples for each node. In the backward pass, gradients for nodes that are directly connected to the cost nodes can be computed with usual backpropagation. For stochastic nodes, we first compute the costs $\hat{C}_v$ for node $v$ in Equation 3.2. Broadcasting these values to the corresponding nodes, we can then continue backpropagation from the stochastic nodes with the gradient of the log probability.

Finally, if $g_\theta$ is the resulting gradient for parameter $\theta$, our gradient descent update is

$$\theta \leftarrow \theta - \eta g_\theta \tag{3.4}$$

where $\eta$ is a learning rate.

**Algorithm 1** Gradient Descent for SCGs

Forward pass through SCG
**for all** $c \in \mathcal{C}$ **do**
  $\hat{c} \leftarrow \hat{c} - b_c$ ▷ Subtract baselines from costs
**end for**
**for all** $v \in$ SCG **do**
  $\mathbf{g}_v = \begin{cases} 1 & \text{if } v \in \mathcal{C} \\ 0 & \text{else} \end{cases}$
  Compute $\hat{C}_v \leftarrow \sum_{c \in \mathcal{C},\, v \prec c} \hat{c}$ ▷ Aggregated costs
**end for**
**for** $v$ in REVERSETOPOLOGICALORDER(SCG) **do** ▷ Backpropagation
  **for** $w \in \mathrm{par}(v)$ **do**
    **if not** ISSTOCHASTIC($w$) **then**
      **if** ISSTOCHASTIC($v$) **then**
        $\mathbf{g}_w$ += $\hat{C}_w \cdot \frac{\partial}{\partial w} \log p(v|\, \mathrm{par}(v))$
      **else**
        $\mathbf{g}_w$ += $(\frac{\partial v}{\partial w})^\top \mathbf{g}_v$
      **end if**
    **end if**
  **end for**
**end for**
**for all** parameters $\theta$ **do**
  $\theta \leftarrow \theta - \eta g_\theta$ ▷ Gradient descent
**end for**

Algorithm 1 gives the complete algorithm for gradient descent on SCGs.

Note that the resulting algorithm can easily be implemented in standard autodifferentiation packages for neural networks (e.g. Tensorflow, Torch). We only need to broadcast the costs $\hat{C}_v$ to each stochastic node, and implement the gradients accordingly.

## 3.4 Relation to Our Models

Now that we understand the general definition of SCGs, we briefly overview how we will use them.

In the NLP problems we are interested in, we have a sequential decision problem as in reinforcement learning. That is, we have a trajectory of states $s_t$ and we make actions $a_t$ at time $t$ based on some parameterized policy function. Each action leads to a reward $r_t$ and determines our next state $s_{t+1}$, thus influencing total future reward $\sum_{s=t}^{T} r_s$ for finite

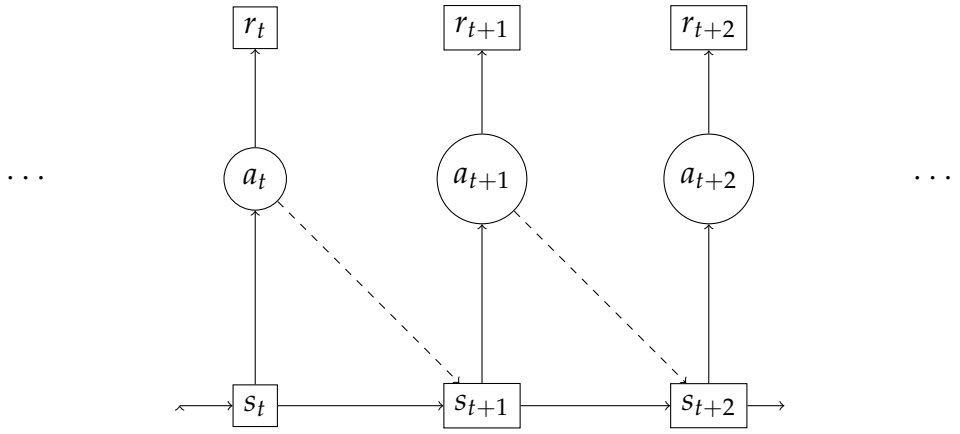



*Figure 3.2: The seq2seq SCG: the general form of the stochastic computation graph that appears in our problems. The squares nodes (states $s_t$) are deterministic hidden states, the circular nodes (actions $a_t$) are stochastic, and $r_t$ comes from our training loss. Our graph is abbreviated — each arrow can potentially have many intermediate nodes and parameters.*

$T$. See Figure 3.2 for a diagram.

The resulting SCG will turn out to mirror the computation graph for sequence-to-sequence models with attention. There, $s_t$ will be the hidden state of our model when generating word $t$, and $r_t$ comes from our loss function during training. In that case, the stochasticity of $a_t$ can arise from a mechanism such as hard attention.

With this setup in mind, we describe our models in detail in the next chapter.

# Chapter 4

# Models

In this chapter, we describe our models. We begin by introducing the standard sequence-to-sequence attention model, then describe our extensions of the basic architecture.

## 4.1 Sequence-to-sequence (seq2seq)

We first describe the neural network architecture of the seq2seq models, also known as encoder-decoder models (Bahdanau et al., 2014).

In the seq2seq model, an *encoder* recurrent neural network (RNN) reads the source sequence as input to produce the *context*, and a *decoder* RNN generates the output sequence using the context as input. One popular RNN choice is the long-short term memory (LSTM) network (Hochreiter & Schmidhuber, 1997).

More formally, suppose we have a vocabulary $\mathcal{V}$. A given input sentence $w_1, \ldots, w_n \in \mathcal{V}$ is transformed into a sequence of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^{d_{in}}$ through a word embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d_{in}}$ as $\mathbf{x}_t = \mathbf{E}w_t$.

An RNN is given by a parametrizable function $f_{enc}$ and a hidden state $\mathbf{h}_t \in \mathbb{R}^{d_{hid}}$ at each time step $t$ with $\mathbf{h}_t = f_{enc}(\mathbf{x}_t, \mathbf{h}_{t-1})$. For the LSTM, we keep an auxiliary state $\mathbf{c}_t$

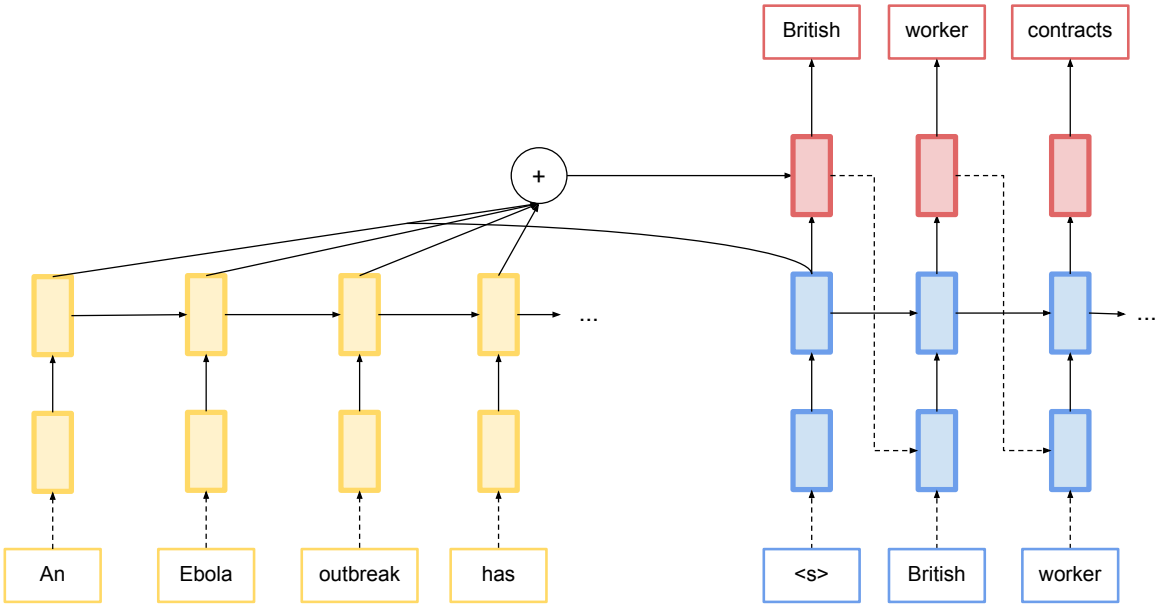



*Figure 4.1: Model architecture for sequence-to-sequence with attention, or* MODEL 0. *The yellow hidden states are the encoder, the blue hidden states are the decoder, and the red hidden states are the generator. The decoder hidden state at each time step determines the attention weights, which we use to average the encoder hidden states to produce a context vector. The result feeds into the generator.*

along with $\mathbf{h}_t$, and we compute $f_{enc}$ as

$$\mathbf{f}_t = \sigma(\mathbf{W}^f \mathbf{x}_t + \mathbf{U}^f \mathbf{h}_t + b_f) \tag{4.1}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}^i \mathbf{x}_t + \mathbf{U}^i \mathbf{h}_t + b_i) \tag{4.2}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}^o \mathbf{x}_t + \mathbf{U}^o \mathbf{h}_t + b_o) \tag{4.3}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}^c \mathbf{x}_t + \mathbf{U}^c \mathbf{h}_{t-1} + b_c) \tag{4.4}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{4.5}$$

where $\mathbf{W}, \mathbf{U}, b$ are learned parameters, $\sigma$ is the sigmoid function, and $\odot$ is the elementwise product. Intuitively, $\mathbf{c}_t$ is the memory cell, $\mathbf{f}_t$ is the forget gate, $\mathbf{i}_t$ is the input gate, $\mathbf{h}_t$ is the output cell, and $\mathbf{o}_t$ is the output gate.

LSTMs can be stacked on top of one another by treating the outputs $\mathbf{h}_t$ of one LSTM as the inputs to another LSTM. In stacked LSTMs, we will take the top sequence of hidden states $\mathbf{h}_1, \ldots, \mathbf{h}_n$ to form the context hidden states.

The decoder is another RNN $f_{dec}$ that generates output words $y_t \in \mathcal{V}$. It keeps hidden state $\mathbf{h}_t^{dec} \in \mathbb{R}^{d_{hid}}$ as $\mathbf{h}_t^{dec} = f_{dec}(y_{t-1}, \mathbf{h}_{t-1}^{dec})$ similar to the encoder RNN. A context vector is produced at each time step using an attention function $a$ that takes the encoded hidden states $[\mathbf{h}_1, \ldots, \mathbf{h}_n]$ and the current decoder hidden state $\mathbf{h}_t^{dec}$ and produces the context $\mathbf{c}_t \in \mathbb{R}^{d_{ctx}}$:

$$\mathbf{c}_t = a([\mathbf{h}_1, \ldots, \mathbf{h}_n], \mathbf{h}_t^{dec}) \tag{4.6}$$

As in Luong et al. (2015), it is helpful to feed the context vector at time $t-1$ back into the decoder RNN at time $t$, i.e. $\mathbf{h}_t^{dec} = f_{dec}([y_{t-1}, \mathbf{c}_{t-1}], \mathbf{h}_{t-1}^{dec})$.

Finally, a linear projection produces a distribution over output words $y_t \in \mathcal{V}$:

$$p(y_t | y_{t-1}, \ldots, y_1, [h_1, \ldots, h_n]) = \mathbf{W}^{out}\mathbf{c}_t + b^{out} \tag{4.7}$$

Given document-summary pairs $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$ for training, the models are trained to maximize the log probability of getting the sequences in the dataset correct, i.e. minimize the negative log-likelihood (NLL):

$$\begin{aligned}
\mathcal{L}(\theta) &= -\sum_{i=1}^{N} \log p(\mathbf{y}^{(i)} | \mathbf{x}^{(i)}; \theta) \\
&= -\sum_{i=1}^{N} \sum_{t=1}^{T-1} \log p(y_{t+1}^{(i)} | \mathbf{x}^{(i)}, y_t^{(i)}, \ldots, y_1^{(i)}; \theta)
\end{aligned}$$

As the model is fully differentiable with respect to its parameters, we can train it end-to-end with stochastic gradient descent and the backpropagation algorithm.

We note that we have great flexibility in how our attention function $a(\cdot)$ combines the encoder context and the current decoder hidden state. In the next few sections, we explain standard choices for $a(\cdot)$ as well as our proposed model of coarse-to-fine attention.

## 4.2 Model 0: Standard Attention

In Bahdanau et al. (2014), the function $a(\cdot)$ is implemented with an *attention network*. We compute attention weights for each encoder hidden state $h_i$ as follows:

$$\beta_{t,i} = \mathbf{h}_i^\top \mathbf{W}^{attn} \mathbf{h}_t^{dec} \quad \forall i = 1, \ldots, n \tag{4.8}$$

$$\boldsymbol{\alpha}_t = \text{softmax}(\boldsymbol{\beta}_t) \tag{4.9}$$

$$\widetilde{\mathbf{c}}_t = \sum_{i=1}^{n} \alpha_{t,i} \mathbf{h}_i \tag{4.10}$$

The idea behind attention is to select the most relevant words of the source (by assigning higher attention weights) when generating output word $y_t$ at time $t$.

The softmax function, defined as

$$\text{softmax}([\beta_1, \ldots, \beta_n])_i = \frac{\exp(\beta_i)}{\sum_{j=1}^{n} \exp(\beta_j)} \tag{4.11}$$

normalizes the $\alpha_i$ to sum to 1 over the source sentence words. This gives us a notion of probability distribution over the encoder words — we can therefore write $\mathbf{c}_t$ as the expectation $\mathbb{E}_\alpha[\mathbf{h}]$, where we pick $\mathbf{h}_i$ with probability $\alpha_i$.

Our final context vector is then

$$\mathbf{c}_t = \tanh(\mathbf{W}^2[\widetilde{\mathbf{c}}_t, \mathbf{h}_t^{dec}]) \tag{4.12}$$

for $\mathbf{W}^2 \in \mathbb{R}^{2d_{hid} \times d_{ctx}}$ a learned matrix.

Going forward, we call this instantiation of the attention function MODEL 0.

## 4.3 Model 1 and 2: Coarse-to-Fine Soft Attention

The attention network of MODEL 0 is computationally expensive for long sentences — for each hidden state of the decoder, we need to compare it to every hidden state of the encoder in order to determine where to attend to. This seems unnecessary for a problem such as document summarization; intuitively, we only need to attend to a few important sentences at a time. Therefore, we propose a hierarchical method of attending to the document by first attending to sentences, then to the words within sentences. We call this

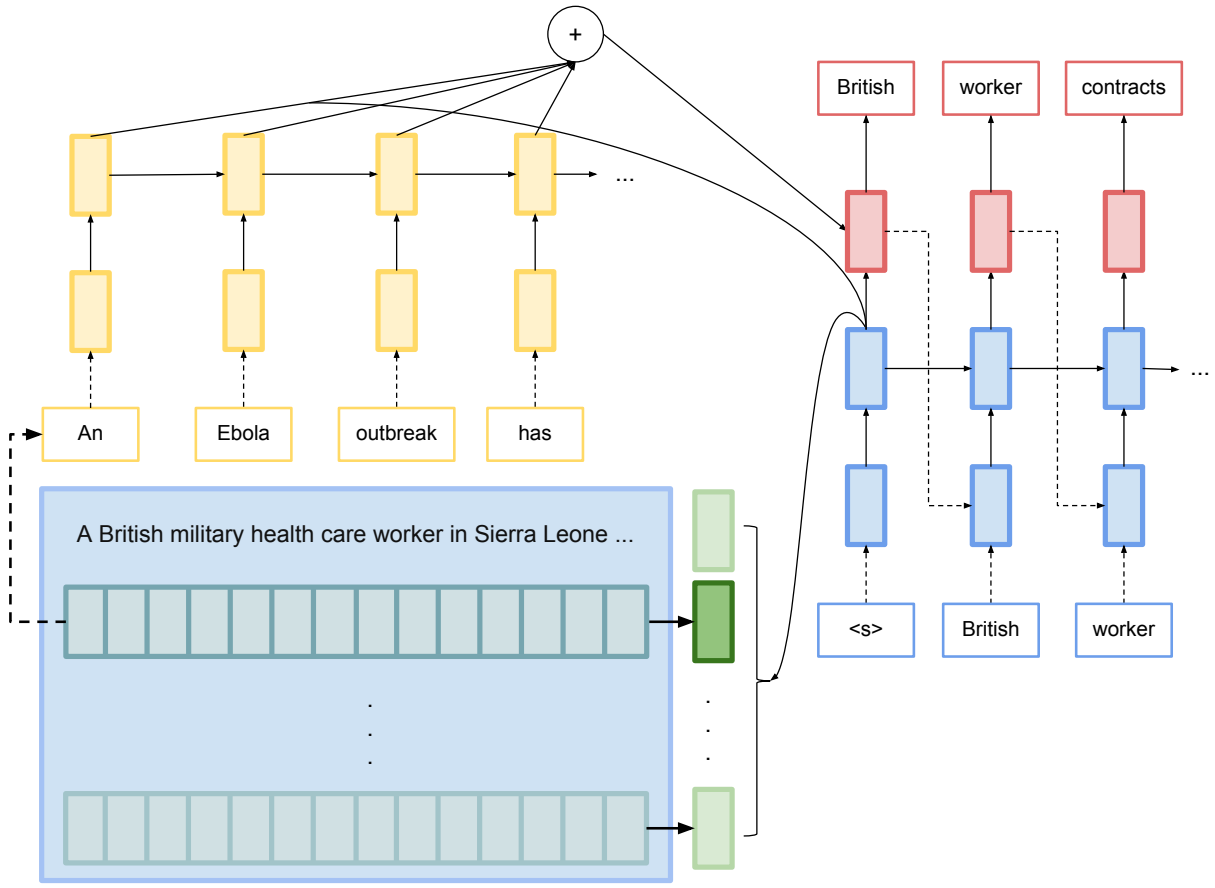



*Figure 4.2: Model architecture for sequence-to-sequence with coarse-to-fine attention. The left side is the encoder that reads the document, and the right side is the decoder that produces the output sequence. On the encoder side, the green hidden states (sentence-level) are used for the coarse attention weights, while the yellow hidden states (word-level) are used for the fine attention weights. The context vector is then produced by averaging the word-level states. In* MODEL 2*, we average over the coarse attention weights, thus requiring computation of all word-level hidden states. In* MODEL 3*, we make a hard decision for which sentence to use, and so we only need to compute word-level hidden states for one sentence.*

method *coarse-to-fine attention*[1].

To be able to attend to both sentences and words in a hierarchical manner, we need to construct encodings of the document at both levels. For the coarse-grained sentence representations, we use a simple encoding model (e.g. bag of words), and for the fine-grained representations, we run an LSTM encoder on the words of a sentence. Thinking about text at different levels of granularity is motivated by Li et al. (2015), who use the idea of a hierarchical representation of text to develop an autoencoder for paragraph representation.

Therefore, if we can make our model first use coarse attention to choose sentences, then use fine attention to choose words only from that sentence, then we avoid the computational cost of searching over the entire document.

Specifically, suppose we have sentences $s_1, \ldots, s_m$ with words $w_{i,1}, \ldots, w_{i,n_i}$ for sentence $s_i$. We apply an RNN to each sentence separately to get corresponding hidden states $\mathbf{h}_{i,j}$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n_i$, so that

$$\mathbf{h}_{i,j} = \text{RNN}(\mathbf{h}_{i,j-1}, w_{i,j}) \tag{4.13}$$

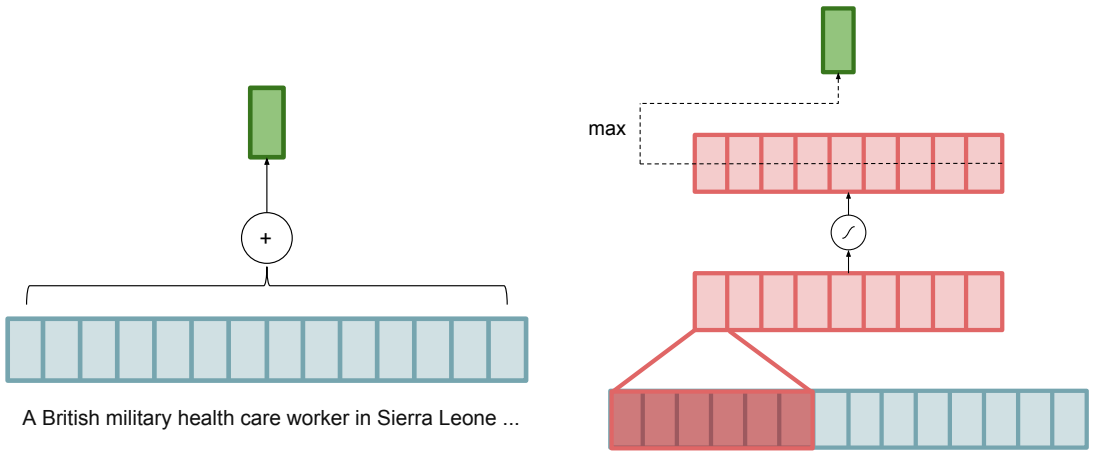
For attention, we then consider two options.

**Model 1** We can follow MODEL 0 and compute attention weights $\alpha_{i,j}$ for each hidden state $\mathbf{h}_{i,j}$ by normalizing over all states. This is a slightly weakened version of MODEL 0; like MODEL 0, we normalize attention over all hidden states at once, but now our RNN hidden states do not share context between sentences. We call this MODEL 1.

**Model 2** Alternatively, rather than taking attention over the entire document, we can instead have a two-layered hierarchical attention mechanism: first, we have weights $\alpha^s_1, \ldots, \alpha^s_m$ for each sentence, and then for sentence $s_i$, we have another set of weights $\alpha^w_{i,1}, \ldots, \alpha^w_{i,n_i}$. Our final attention weight on word $w_{i,j}$ is then $\alpha_{i,j} = \alpha^s_i \cdot \alpha^w_{i,j}$.

In order to compute the sentence attention weights $\alpha^s_i$, we need to produce representations of each sentence; i.e., given the words $w_{i,1}, \ldots, w_{i,n_i}$ of the sentence, we produce a vector representation $\mathbf{h}^s_i \in \mathbb{R}^{d_{sent}}$. Figure 4.3 shows diagrams of our sentence representation models.

[1]The term coarse-to-fine attention has previously been introduced in the literature (Mei et al., 2016). However, their idea is different: they use coarse attention to reweight the fine attention computed over the entire input. This idea has also been called hierarchical attention (Nallapati et al., 2016).




*(a) Bag-of-words*

*(b) Convolution*

*Figure 4.3: Options for producing sentence representations from the word embeddings. We can either use a bag of words by summing the embeddings, or apply convolutions and max-over-time pooling. The output is a sentence representation in* $\mathbb{R}^{d_{sent}}$.

Our first option is bag of words: we simply take the representation as

$$\mathbf{h}_i^s = \sum_{j=1}^{n_i} \mathbf{E} w_{i,j} \tag{4.14}$$

i.e. the sum of the word embeddings, where $\mathbf{E}$ is another embedding matrix.

Alternatively, we can use a convolutional method: as in Kim (2014), we perform a convolution over each window of words in the sentence using a fixed kernel width. We use max-over-time pooling to obtain a fixed-dimensional sentence representation in $\mathbb{R}^{d_f}$ where $d_f$ is the number of filters.

Explicitly, fix a sentence and suppose we have word vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with $\mathbf{x}_j = \mathbf{E} w_j \in \mathbb{R}^{d_{in}}$, and suppose we have kernel width $k$ and convolution weights $\mathbf{W}^{conv} \in \mathbb{R}^{d_f \times d_{in} k}$ where $d_f$ is the number of filters. Then applying the convolution $\mathbf{W}^{conv} * [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ gives result $\mathbf{u} = [\mathbf{u}_1, \ldots, \mathbf{u}_{n-k+1}]$ with $j$th element

$$\mathbf{u}_j = \mathbf{W}^{conv} \cdot [\mathbf{x}_j, \mathbf{x}_{j+1}, \ldots, \mathbf{x}_{j+k-1}] + \mathbf{b}^{conv} \in \mathbb{R}^{d_f}$$

Our final output is given by

$$\mathbf{h}^s = \max_j(\tanh(\mathbf{u}_j)) \tag{4.15}$$

where the max-over-time takes the maximum along the word indexing dimension.

As an addition to any sentence representation method, we can include *positional embeddings*. In general, we expect the order of sentences in the document to matter for summarization — for example, the first few sentences are usually important. We therefore include the option to concatenate a small fixed-dimensional embedding of the sentence's position to the existing representation.

Thus, using the sentence representations, we can compute attention $\alpha_i^s$ over the sentences. For the words in each sentence, we run an LSTM over each sentence separately, and create attention weights $\alpha_{i,j}^w$ over the words in each sentence in the same way as MODEL 0. Using attention on word $w_{i,j}$ as $\alpha_{i,j} = \alpha_i^s \cdot \alpha_{i,j}^w$, we can proceed exactly as in MODEL 0 by computing the weighted average over hidden states $\mathbf{h}_{i,j}$.

We call this method of attention MODEL 2.

## 4.4 Model 3: Coarse-to-Fine Sparse Attention

With the previous models, we are required to compute hidden states over all words and sentences in the document, so that if we have $M$ sentences and $N$ words per sentence, the computational complexity is $O(MN)$ for each attention step.

However, if we are able to perform conditional computation and only compute on $M^+$ of the sentences, we can reduce the complexity to $O(M^+N)$. If we are able to make $M^+$ constant or even 1, the complexity of attention becomes $O(N)$ — an expression invariant of the length of the document!

In our experiments, we will apply stochastic sampling to the attention distribution $\boldsymbol{\alpha}$ in the spirit of "hard attention" (Xu et al., 2015). Specifically, rather than computing the context $\widetilde{\mathbf{c}}$ as an expectation over $\boldsymbol{\alpha}$ (i.e. $\widetilde{\mathbf{c}} = \sum_{i=1}^n \alpha_i \mathbf{h}_i$), we can sample from the probability distribution $\boldsymbol{\alpha}$ to obtain a single state $\mathbf{h}_i$, and we set $\widetilde{\mathbf{c}} = \mathbf{h}_i$ as the sampled hidden state.

In our case, we take MODEL 2 and apply hard attention at the sentence level, but keep the word level attention per sentence as is. That is, we sample from the attention weights $\alpha_1^s, \ldots, \alpha_m^s$ to obtain a one-hot encoding for the sentence attention, and apply the same multiplication with this one-hot vector on the word-level attention weights $\alpha_{i,1}^w, \ldots, \alpha_{i,n_i}^w$ for all $i = 1, \ldots, m$. At test time, we take the max $\alpha_i^s$ for our one-hot encoding instead of

sampling. We call this MODEL 3.

Because the hard attention model loses the property of being end-to-end differentiable, we use reinforcement learning to train our network.

### 4.4.1 Practical Considerations

Our hard attention model is now a stochastic computation graph, and thus Algorithm 1 from Section 3.3 applies.

Here, we have an RL agent where the state $s_t$ is the LSTM decoder state at time $t$, and actions $a_t$ are the hard attention decisions. Since samples from $\boldsymbol{\alpha}_t$ at time $t$ of the RNN decoder can also affect future rewards, in the notation of Chapter 3, the total influenced reward is $\hat{C}_t = \sum_{s=t}^{T} r_s$ at time $t$, where $r_t = \log p(y_t|y_1, \ldots, y_{t-1}, \mathbf{x})$ is the single step reward.

Inspired by the discount factor from RL, we slightly modify the total reward: instead of simply taking the sum, we can scale later rewards with a discount factor $\gamma$, giving

$$\hat{C}_{a_t} = \sum_{s=t}^{T} \gamma^{s-t} r_s \tag{4.16}$$

for the stochastic hard attention node $a_t$. We found that adding a discount factor helps in practice (we use $\gamma = 0.5$).

To calculate the baselines for variance reduction, we store a constant $b_t$ for each decoder time step $t$. We follow Xu et al. (2015) and keep an exponentially moving average of the reward for each time step $t$:

$$b_t \leftarrow b_t + \beta(r_t - b_t) \tag{4.17}$$

where $r_t$ is the average minibatch reward and $\beta$ is a learning rate (set to 0.1).

While several papers suggest using a learned baseline (e.g. Mnih et al., 2014; Ranzato et al., 2015), we have not found this to be effective. In our experiments, we found that attempting to learn the baseline failed to converge, most likely because there is not enough correlation between the reward and the hidden states preceding the attention layer.

In addition to including a baseline, we also scale the rewards by a tuned hyperparameter — we found that scaling helped to stabilize training. We empirically set the scale to 0.3.

**ALTERNATE training** Xu et al. (2015) explain that training hard attention with REINFORCE has very high variance, even when including a baseline. Thus, for every minibatch of training, they randomly use soft attention instead of hard attention with some probability (they use 0.5). The backpropagated gradient is then the standard soft attention gradient instead of the REINFORCE gradient. In our results, we label this as +ALTERNATE.

While this method helps stabilize training, we would prefer not to use it. Our goal in coarse-to-fine attention is to reduce computation, but ALTERNATE training requires full attention computation for a subset of minibatches. However, this training method still allows for computational benefits at test time, and we include it in our experiments to test the feasibility of hard attention.

**Multiple samples** From our initial experiments with MODEL 3, we found that taking a single sample was not very effective. However, we discovered that sampling multiple times from the attention distribution $\boldsymbol{\alpha}^s$ improves performance.

To be precise, we fix a number $k_{mul}$ for the number of times we sample from $\boldsymbol{\alpha}^s$. Then, we sample based on the multinomial distribution $\boldsymbol{\mu} \sim \text{Mult}(k_{mul}, \{\alpha_i\}_{i=1}^m)$ to produce the new sentence-level attention vector $\widetilde{\boldsymbol{\alpha}}^s$, with $\widetilde{\alpha}_i^s = \mu_i / k_{mul}$. In our results, we label this as +MULTI.

Intuitively, $k_{mul}$ is the number of sentences we select to produce the context. With higher $k_{mul}$, the hard attention model more closely approximates the soft attention model, and hence leads to better performance. This, however, incurs a cost in computational complexity.

# Chapter 5

# Experiments

In this chapter, we describe our experimental setup.

## 5.1 Data

### 5.1.1 CNN/Dailymail

Experiments were performed on the CNN/Dailymail dataset from Hermann et al. (2015). While the dataset was created for a question-answering task, the dataset format is suited for summary. Each data point is a news document accompanied by up to 4 "highlights", and we take the first of these as our target summary.

Train, validation, and test splits are provided along with document tokenization and sentence splitting. We do additional preprocessing by replacing all numbers with # and appending end of sentence tokens <s> to each sentence. We limit our vocabulary size to the 50000 most frequent words, replacing the rest with <unk> tokens. We dropped the documents which had an empty source (which came from photo articles).

Table 5.1 lists statistics for the CNN/Dailymail dataset. Figure 5.1 shows examples source and target pairs from the dataset.

In the context of these new datasets, the summarization task has not yet been fully standardized. Research in the area is still largely preliminary, with only a few papers reporting results (e.g. Nallapati et al., 2016). While CNN/Dailymail may not be the most suitable dataset for the task due to its noisiness (Chen et al., 2016), a better alternative is yet to exist.

| Document | Summary |
|---|---|
| ( cnn ) the man suspected of killing a deputy u.s. marshal at a motel in baton rouge , louisiana , has died , brittany stewart in the east baton rouge coroner 's office said wednesday . </s> the cause of death is pending autopsy , she said . </s> jamie croom , ## , was wounded in a shootout with deputy u.s. marshal josie wells . </s> it can be one of the most dangerous tasks for a law enforcement officer : serving an arrest warrant to a fugitive murder suspect . when wells tried to do that tuesday , he lost his life . </s> ... | the fugitive who killed the marshal was " extremely dangerous , " u.s. marshals service director says |
| ( cnn ) there have been a few times in my career when i 've been thoroughly disappointed – even disgusted – with my fellow women in the workplace . </s> no , i certainly do n't expect all my female colleagues to go out of their way for me and sing " kumbaya " together in the office , but i 'm always stunned when a woman who could have been helpful to me was n't , when a woman who could have been a mentor chose not to be , when a woman tried to hurt me because of her own fear , anxiety or what have you . </s> i 'd love to say more about each of the women i 've met along the way who fit those descriptions , but my point is not to single anyone out . my goal is to ask the question , " why ? " </s> obviously , not all women are like this and there are plenty of men guilty of the same behavior , but why do so many women try to tear each other down instead of lift each other up ? </s> ... | cnn 's kelly wallace wonders why women too often do n't lift each up in the workplace |
| much of the start of the world 's most famous sled dog race is covered in barren gravel , forcing iditarod organizers to move the start further north where there is snow and ice . </s> a weather pattern that buried the eastern u.s. in snow has left alaska fairly warm and relatively snow - free this winter , especially south of the alaska range . </s> ' if i have one more person say to me to move the iditarod to boston , i 'm going to shake my head , ' said race director mark nordman . </s> scroll down for video in this photo taken on thursday , there are bare patches of grass and mud on sled dog trails in anchorage , alaska which is unsuitable for the iditarod </s> ... | much of the start of the world 's most famous sled dog , the iditarod trail sled dog race , is covered in barren gravel |

*Figure 5.1: Examples of source and target for the CNN/Dailymail dataset. Data is shown after preprocessing. In the first example, the summary is from a quote later on in the document. The second example is similar, but the start of the document has low information content. In the third, the summary is almost identical to the first sentence. See Appendix A for the full documents.*

| Dataset | CNN | Dailymail | Combined |
|---|---|---|---|
| Train size | 90266 | 196961 | 287227 |
| Valid size | 1220 | 12148 | 13368 |
| Test size | 1093 | 10397 | 11490 |
| Avg. # words per doc. | 794 | 831 | 819 |
| Avg. # sent. per doc. | 21 | 29 | 26 |
| Avg. # words per sent. | 36 | 27 | 29 |
| Avg. # words per summary | 13 | 14 | 14 |

*Table 5.1: Statistics for CNN/Dailymail data. Numbers are collected over the training data.*

## 5.2 Implementation Details

A few implementation details were necessary to make minibatch training possible. First, instead of taking attention over each individual sentence, we arrange the first 400 words of the document into a 10 by 40 image, and take each row to be a sentence.

Second, we pad short documents to the maximum length with a special padding word, and allow the model to attend to it. However, we zero out word embeddings for the padding states and also zero out their corresponding LSTM states. We found in practice that very little of the attention ended up on the padding words.

Ideally, we would prefer to not truncate documents, but GPU memory limits the number of words we can use. However, we believe that this usually does not matter as the average document length is about 800 words, and half of that should be sufficient context to summarize. Nonetheless, this should be explored in future work.

## 5.3 Models

**Baselines** For a baseline, we take the first sentence of the document. We call this FIRST.

We also consider the feature-based document summarizer of Durrett et al. (2016), which uses integer linear programming (ILP) methods to compress extracted sentences. We apply the code [1] directly on the test set without retraining the system. Their system requires that the documents are preprocessed in CONLL format, so we use the Berkeley coreference system[2] with the coreference and NER settings. We call this baseline BERKELEY.

[1] https://github.com/gregdurrett/berkeley-doc-summarizer
[2] https://github.com/gregdurrett/berkeley-entity

**Our models** We ran experiments with Models 0 to 3 as described above.

- MODEL 0: Sequence-to-sequence model.
- MODEL 1: LSTM encoder per sentence, soft attention over all.
- MODEL 2: Coarse-to-fine with soft attention.
- MODEL 3: Coarse-to-fine with hard attention over sentences.

We use convolutions for the coarse attention by default, and we write BOW when we use bag-of-words instead. For BOW models, we fix the word embeddings on the encoder side (in other models, they are fine tuned). We also include the option of including positional embeddings for sentence representations, which we denote as +POS.

For MODEL 3, we include options +MULTI for $k_{mul} > 1$, +PRETRAIN for starting with a model pretrained with soft attention for 1 epoch, and +ALTERNATE for sampling between hard and soft attention with probability 0.5.

For MODEL 2, our default document arrangement is a 10 by 40 grid of words. We also experiment with shapes of 5 by 80 and 2 by 200 (denoted 5X80, 2X200 resp.). These should more closely approximate MODEL 0 as the shape approaches a single sequence.

## 5.4 Training

We train with minibatch stochastic gradient descent (SGD) with batch size 20 for 20 epochs, renormalizing gradients below norm 5. We initialize the learning rate to 0.1 for the sentence encoder and 1 for the rest of the model, and begin decaying it by a factor of 0.5 each epoch after the validation perplexity stops decreasing.[3]

We use 2 layer LSTMs with 500 hidden units, and we initialize word embeddings with 300-dimensional word2vec embeddings (Mikolov & Dean, 2013). We initialize all other parameters as uniform in the interval $[-0.1, 0.1]$. For convolutional layers, we use a kernel width of 6 and 600 filters. Positional embeddings have dimension 25. We use dropout (Srivastava et al., 2014) between stacked LSTM hidden states and before the final word generator layer to regularize (with dropout probability 0.3).

[3]We tried more complicated SGD optimization methods such as Adagrad (Duchi et al., 2011) and Adam (Kingma & Ba, 2015), but found that they did not perform as well. This could be due to gradient norms that are too large.

At test time, we run beam search to produce the summary with a beam size of 5. This is necessary as computing the full joint probability of outputs is computationally intractable.

Our models are implemented using Torch (Collobert et al., 2011a) based on a past version of Harvard's OpenNMT system[4]. We ran our experiments on a 12GB Geforce GTX Titan X GPU. The models take between 2-2.5 hours to train per epoch.

All of our code is available open source[5].

In the next chapter we show results.

[4]https://github.com/harvardnlp/seq2seq-attn
[5]https://github.com/jeffreyling/seq2seq-hard

# Chapter 6

# Results

## 6.1 Evaluation

We report metrics for perplexity and ROUGE scores (Lin, 2004) on the test set. We use the trained models with the best validation perplexity.

Perplexity is the exponential of the negative log-likelihood, so that smaller perplexity is better (with a lower bound of 1.0).

ROUGE-n computes n-gram overlap between a gold summary and a predicted summary, and ROUGE-L computes the longest common subsequence. We use ROUGE balanced F-scores and report numbers for ROUGE-1 (unigrams), ROUGE-2 (bigrams), and ROUGE-L. While ROUGE traditionally uses the recall metric, we choose F-score since recall is biased towards longer predicted sentences.

With multiple gold summaries in the CNN/Dailymail highlights, we choose to take the max ROUGE score over the gold summaries for a predicted summary, as our models are trained to produce a single sentence. The final metric is then the average over all test data points.

## 6.2 Analysis

Table 6.1 shows summarization results. We see that our soft attention models comfortably beat the baselines, while hard attention falls slightly behind. For the most part, we see that PPL and ROUGE are properly correlated.

The BERKELEY model ROUGE scores are surprisingly low. We attribute this due to the fact that our models usually produce a single sentence as the summary, while the ILP

| Model | PPL | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|
| FIRST | - | 32.3 | 15.5 | 27.4 |
| BERKELEY | - | 29.1 | 16.0 | 26.5 |
| MODEL 0 | 13.9 | 34.7 | 18.8 | 32.3 |
| MODEL 1 | 14.9 | 33.0 | 17.5 | 30.7 |
| MODEL 2 | 16.0 | 33.3 | 17.5 | 31.0 |
| MODEL 2 BOW | 16.3 | 33.0 | 17.4 | 30.7 |
| MODEL 2 +POS | 15.4 | 34.2 | 18.3 | 31.8 |
| MODEL 2 5X80 | 15.0 | 33.9 | 18.0 | 31.5 |
| MODEL 2 2X200 | 14.5 | 33.9 | 18.1 | 31.6 |
| MODEL 3 | 32.8 | 28.2 | 12.9 | 26.2 |
| MODEL 3 +POS | 37.8 | 28.3 | 12.5 | 26.1 |
| MODEL 3 +MULTI2 | 25.5 | 30.0 | 14.4 | 27.9 |
| MODEL 3 +POS +MULTI2 | 21.9 | 31.2 | 15.3 | 29.0 |
| MODEL 3 +MULTI3 | 22.9 | 30.4 | 14.9 | 28.3 |
| MODEL 3 +PRETRAIN | 26.3 | 29.7 | 14.2 | 27.5 |
| MODEL 3 +ALTERNATE | 23.6 | 31.1 | 15.4 | 28.8 |

*Table 6.1: Summarization results for CNN/Dailymail on the test set. The ROUGE numbers are balanced F-scores. Lower PPL is better, higher ROUGE is better.*

system can produce multiple. The BERKELEY model therefore has comparatively high ROUGE recall while suffering in precision.

Unfortunately, the MODEL 0 sequence-to-sequence baseline proves to be difficult to beat. MODEL 1 performs surprisingly poorly in ROUGE, despite its good perplexity.

MODEL 2 has worse performance, likely due to our assumption that we can factor the attention distribution into a coarse distribution and a fine distribution. This assumption is quite strong — we hypothesize that our deficit then exists because either (1) our sentence representations are not sufficiently strong, or (2) we did not properly solve the optimization problem. We believe (1) may be true because with perfect modeling power, the model should be able to learn accurate representations of sentences such that it can attend to the right ones. We believe (2) is also an issue because the training signal is backpropagated to the word-level LSTM via the attention weights — since the training algorithm cannot directly compare word attention weights as in MODEL 0 or MODEL 1, it has trouble finding the best optimum. Additionally, we found empirically that Adagrad and Adam performed worse than SGD, which is unusual given that these methods are

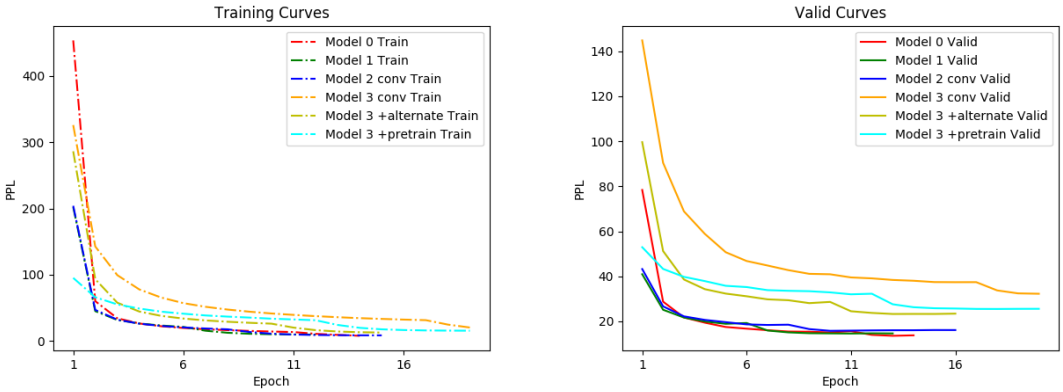



*Figure 6.1: Training and validation curves for models. Note that* MODEL 3 *takes much longer to converge than the others, and also converges to a worse local optimum.*

usually state-of-the-art.

We observe that MODEL 3 results are significantly worse than soft attention results. This is most likely due to a noisy gradient; even by using variance reduction techniques, the variance of the gradient estimator remains high. It may be too difficult to train the encoder (which forms a large part of the model) using such a noisy gradient. Even with soft attention pretraining (+PRETRAIN) and alternating training (+ALTERNATE), MODEL 3 fails to reach MODEL 2 performance.

While taking a single sample performs quite poorly, we see that taking more than one sample gives a significant boost to scores (+MULTI2, +MULTI3). There seem to be diminishing returns as we take more samples.

Finally, we note that positional embeddings (+POS) give a nontrivial boost to scores (except for MODEL 3 +POS). This makes sense since the position of the sentence in the document should be relevant for determining its importance — for example, we would expect sentences near the start of the document to be more useful for summarization. In the exception, we believe this may be due to difficulty in training the embeddings.

### 6.2.1 Training Curves

We examine training curves for our models in Figure 6.1. We see that hard attention takes longer to converge on average, which likely is due to the noisy gradient. If we include +ALTERNATE, MODEL 3 training converges faster, and if we include +PRETRAIN, the model reaches a better local optimum.

Note also that train curves are well correlated with valid curves. Therefore, we see no

| Model | Entropy |
|---|---|
| MODEL 0 | 1.31 |
| MODEL 1 | 1.58 |
| MODEL 2 | 2.14 |
| MODEL 2 BOW | 2.12 |
| MODEL 2 +POS | 2.06 |
| MODEL 3 | 0.15 |
| MODEL 3 +MULTI2 | 0.59 |
| MODEL 3 +POS +MULTI2 | 0.46 |

*Table 6.2: Entropy over sentence attention, averaged over all attention distributions in the validation set. Lower entropy means a more concentrated distribution. For reference, uniform attention in our case gives entropy* $\approx 2.30$.

signs of overfitting in our models.

### 6.2.2 Entropy

We investigate the entropy of the sentence attention on the validation set in Table 6.2. Entropy for a discrete random variable $Z$ is computed as

$$H(Z) = -\sum_{z} p(Z = z) \log p(Z = z)$$

Intuitively, higher entropy means the attention is more spread out, while lower entropy means the attention is concentrated.

We compute the entropy numbers by averaging over all generated words in the validation set. Because each document has 10 sentences, perfectly uniform entropy would be $\approx 2.30$.

We see that the entropy of MODEL 0 attention is quite low, suggesting that a factorization of attention into sentence-level and word-level is reasonable. However, our MODEL 2 results have very high entropy — it seems as if it uses attention to essentially average all of the encoder hidden states. We examine this more carefully in Section 6.2.4.

Finally, we note that the entropy of MODEL 3 is very low (before taking the argmax at test time). This is exactly what we had hoped for — we will see that the model in fact learns to focus on only a few sentences of the document over the course of generation. If we have multiple samples with +MULTI2, the model is allowed to use 2 sentences at a time, and so it relaxes the entropy slightly.

### 6.2.3 Predicted Summaries

We show some predicted summaries from the model in Figures 6.2, 6.3, and 6.4. The source documents are truncated in these examples for clarity — see Appendix A for the full text.

We note that the BERKELEY system, which extracts sentences first, produces long summaries. In contrast, the generated summaries tend to be quite succint, and most are the result of copying or paraphrasing specific sentences. Clearly, the summaries can usually get the basic gist of the articles.

While the summaries are mostly grammatical, they are occasional redundant (e.g. "the message was posted on twitter and posted on twitter") or have very little information content (e.g. "new : death toll rises to ## , policy say"). In these cases, the model is most likely just sampling from the decoder language model without using the encoder very much.

It is difficult to definitively say which models output the best summaries from this small sample, illustrating the ambiguity of summary metrics. Anecdotally, we note that the models with higher ROUGE scores seem to have qualitatively more grammatical output.

| Model | |
|---|---|
| Source | jasmine coleman , ## , has been found safe and well some ## miles from her home a ## - year - old girl who went missing from her family home at #am amid fears she was driven away by an ' older man ' has been found safe and well . jasmine coleman was reported as missing this morning after disappearing from her home in <unk> , west sussex . the child was found this afternoon following a police appeal some ##miles away in croydon , south east london ... |
| GOLD | jasmine coleman disappeared from her home at around #am this morning |
| FIRST | jasmine coleman , ## , has been found safe and well some ## miles from her home a ## - year - old girl who went missing from her family home at #am amid fears she was driven away by an ' older man ' has been found safe and well . |
| BERKELEY | Jasmine Coleman , 12 , has been found safe and well some 50 miles from her home . A 12-year-old girl has been found safe and well . Jasmine Coleman was reported as missing this morning . The force launched a public appeal for information on Saturday morning . |
| MODEL 0 | jasmine coleman disappeared from her home in <unk> , west sussex |
| MODEL 1 | jasmine coleman , ## , disappeared from home in <unk> , west sussex |
| MODEL 2 | jasmine coleman , ## , was found safe and well in croydon |
| MODEL 2 BOW | jasmine coleman , ## , has been found safe and well in croydon |
| MODEL 2 +POS | jasmine coleman disappeared from her home in <unk> , west sussex |
| MODEL 3 | jasmine coleman , ## , disappeared from family home in <unk> , south east sussex |
| MODEL 3 +MULTI2 | jasmine coleman , ## , disappeared from her home in <unk> , west sussex |
| MODEL 3 +POS +MULTI2 | jasmine coleman disappeared from her home in <unk> , west london |

*Figure 6.2: Predicted summaries for each model, given a short document (an easier example).*

| Model | |
|---|---|
| Source | isis supporters have vowed to murder twitter staff because they believe the site 's policy of shutting down their extremist pages is a ' virtual war ' . </s> a mocked - up image of the site 's founder jack dorsey in <unk> was posted yesterday alongside a diatribe written in arabic , which claimed twitter employees ' necks are ' a target for the soldiers of the caliphate ' . </s> addressing mr dorsey personally , it claimed twitter was taking sides in a ' media war ' which allowed ' slaughter ' , adding : ' your virtual war on us will cause a real war on you . </s> diatribe : an image of twitter founder jack dorsey in <unk> was posted alongside a rant in arabic </s> ... |
| GOLD | diatribe in arabic posted anonymously yesterday and shared online |
| FIRST | isis supporters have vowed to murder twitter staff because they believe the site 's policy of shutting down their extremist pages is a ' virtual war ' . |
| BERKELEY | ISIS supporters have vowed to murder Twitter staff because they believe the site 's policy of shutting down their extremist pages is a ' virtual war ' . Twitter was taking sides . Islamic State militants have swept through huge tracts of Syria and Iraq , murdering thousands of people . |
| MODEL 0 | image of jack dorsey 's founder jack dorsey posted on twitter |
| MODEL 1 | a mocking - up image of jack dorsey was posted on twitter |
| MODEL 2 | the message was posted in arabic and posted on twitter |
| MODEL 2 BOW | the message was posted on twitter and posted on twitter |
| MODEL 2 +POS | dorsey in <unk> was posted yesterday alongside a diatribe in arabic |
| MODEL 3 | ' lone war ' is a ' virtual war ' image of the islamic state |
| MODEL 3 +MULTI2 | isis supporters say site 's policy of shutting down is a ' propaganda war ' |
| MODEL 3 +POS +MULTI2 | twitter users say they believe site 's policy of closure is a ' media war ' |

*Figure 6.3: Predicted summaries for each model, given a longer document.*

| Model | |
|---|---|
| Source | ( cnn ) the man suspected of killing a deputy u.s. marshal at a motel in baton rouge , louisiana , has died , brittany stewart in the east baton rouge coroner 's office said wednesday . </s> the cause of death is pending autopsy , she said . </s> jamie croom , ## , was wounded in a shootout with deputy u.s. marshal josie wells . </s> it can be one of the most dangerous tasks for a law enforcement officer : serving an arrest warrant to a fugitive murder suspect . when wells tried to do that tuesday , he lost his life . </s> ... |
| GOLD | the fugitive who killed the marshal was " extremely dangerous , " u.s. marshals service director says |
| FIRST | ( cnn ) the man suspected of killing a deputy u.s. marshal at a motel in baton rouge , louisiana , has died , brittany stewart in the east baton rouge coroner 's office said wednesday . |
| BERKELEY | -LRB- CNN -RRB- The man suspected of killing a deputy U.S. marshal at a motel in Baton Rouge , Louisiana , has died . Enforcement partners face untold dangers every day in the pursuit of justice , " " The fugitive who killed Deputy Wells was extremely dangerous . |
| MODEL 0 | the cause of death is pending autopsy , she says |
| MODEL 1 | the man suspected of killing a deputy u.s. marshal at a motel in louisiana |
| MODEL 2 | new : u.s. marshals service director <unk> a. <unk> died in baton rouge , louisiana |
| MODEL 2 BOW | the man suspected of killing deputy u.s. marshal killed deputy u.s. marshal |
| MODEL 2 +POS | the cause of death is pending autopsy |
| MODEL 3 | new : body of suspect found in baton rouge , louisiana |
| MODEL 3 +MULTI2 | new : death toll rises to ## , police say |
| MODEL 3 +POS +MULTI2 | the man was killed in baton rouge , louisiana |

*Figure 6.4: Predicted summaries for each model, given a more difficult document.*

### 6.2.4 Attention Heatmaps

For the document in Figure 6.3, we visualize the attention distributions produced by each model in Figures 6.5 to 6.12.

In each figure, the rows are the sentences of each document (40 words per row), and the columns are the summary words produced by the model. The intensity of each box for a given column represents the strength of the attention weight on that row. In particular, the column weights should sum to 1. For MODEL 0 and MODEL 1, heatmaps are produced by summing the word-level attention weights in each row.

We limit our figures to sentence level attention — the full word-level attention is difficult to examine due to the length of the documents.

We see that the spread of the attention matches what we expect to see from the entropy numbers. In MODEL 0 and MODEL 1, the model mostly attends to the right sentences at the right time — for example, in MODEL 1, it copies "a mocked-up image" (with a change of word choice!), then proceeds to get "jack dorsey" from later in the document.

On the other hand, in MODEL 2, the attention becomes washed out with much higher entropy. It seems that the model is essentially averaging all of the encoder hidden states instead of intelligently selecting the right ones. Nonetheless, there are parts of the document that seem slightly more heavily weighted than others.

If we examine the word-level attention (not pictured here), we find that the model focuses on stop words (e.g. punctuation marks, </s>) in the encoder. We posit this may be due to the LSTM "saving" information at these words, and so the soft attention model can best retrieve the information by averaging over these hidden states.

In MODEL 3, we see that we get very sharp attention on some sentences, as we had hoped. Unfortunately, for this example, the model has trouble deciding where to attend to, and oscillates between the first and second-to-last sentence. This seems to be one of the main weaknesses of hard attention models — in case there is not clearly a single right sentence to pick to produce the summary, the model may become indecisive and spit out garbage.

We partially alleviate this problem by allowing the model to attend to multiple sentences in hard attention. Indeed, with +MULTI2, the model actually produces a very coherent output by focusing attention on the first sentence — we also see that it considered choosing the second-to-last sentence and decided against it. We believe that the improved result for this example is not only due to more flexibility in where to attend, but a better encoding model due to the training process (i.e. a better local optimum).

Finally, we observe that +POS encourages the model to attend to sentences closer to the beginning of the document. This matches our intuition that the front of a news article has more information content. This is taken to an extreme by MODEL 3 +POS, which only attends to the first sentence — we believe that during training, the model becomes heavily biased towards attending to the first sentence, and fails to explore the remaining ones.

Appendix B contains a few more attention visualizations.

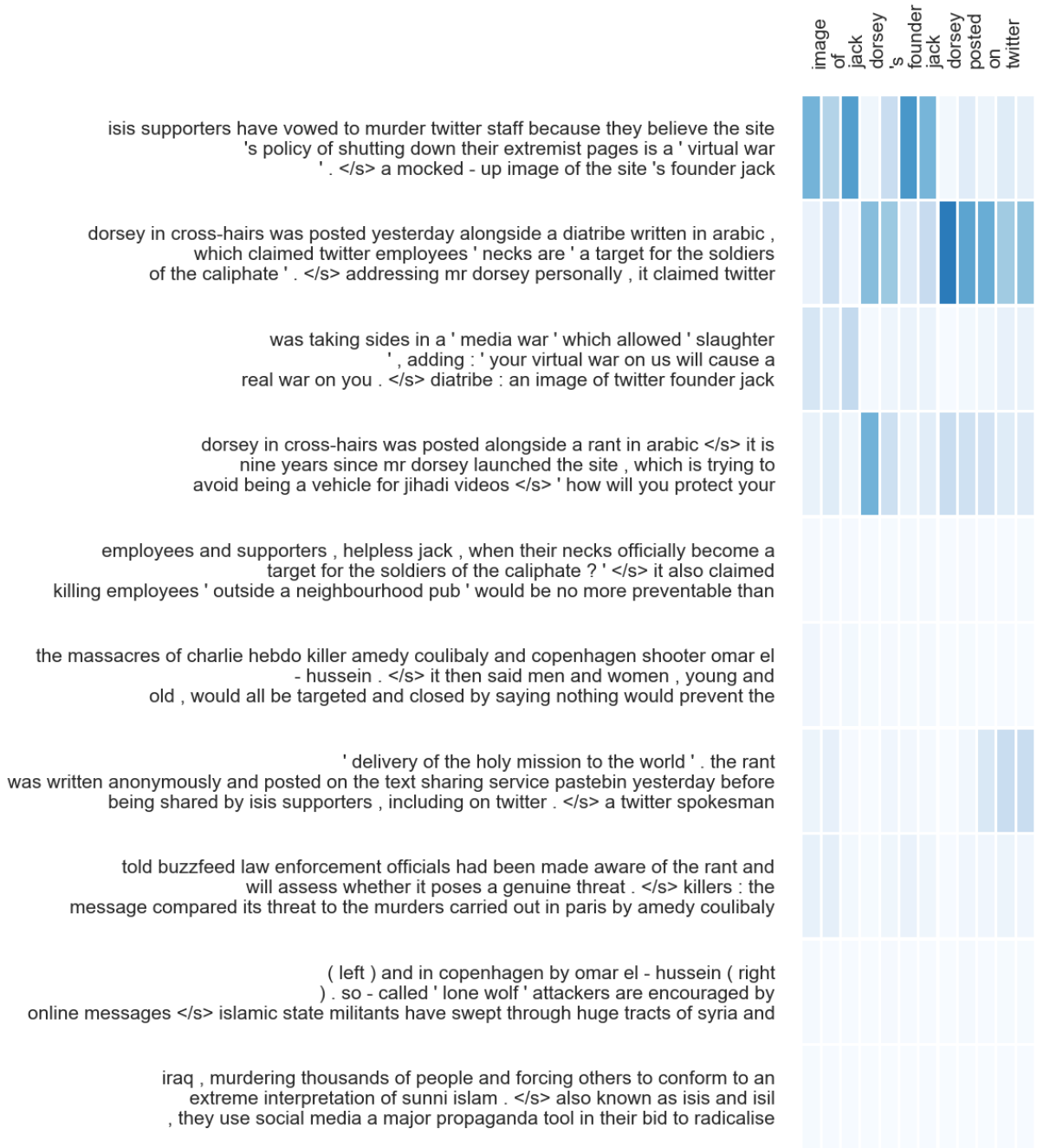



*Figure 6.5:* MODEL 0

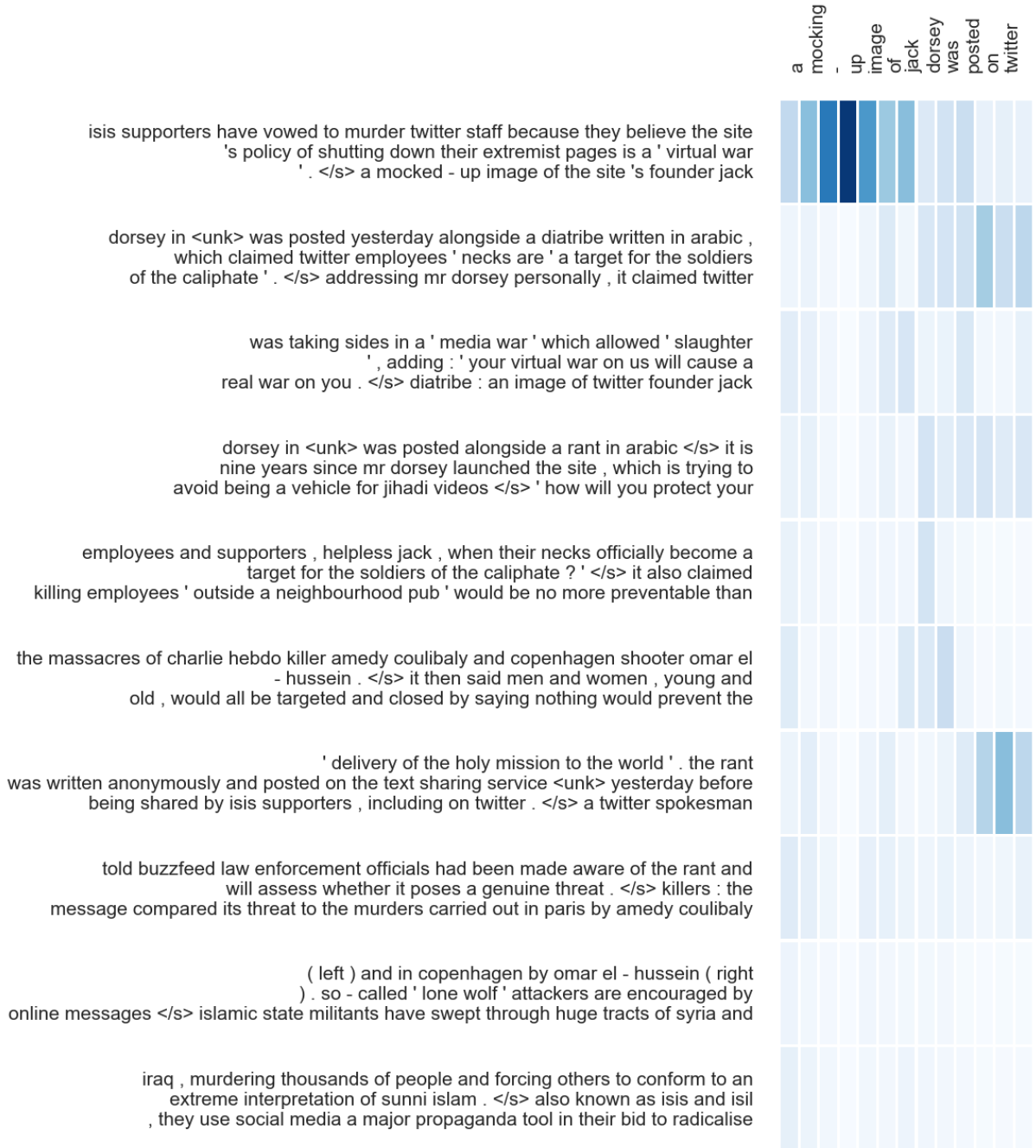



*Figure 6.6:* MODEL 1


*Figure 6.7:* MODEL 2


*Figure 6.8:* MODEL 2 +POS


*Figure 6.9:* MODEL 3


*Figure 6.10:* MODEL 3 +POS

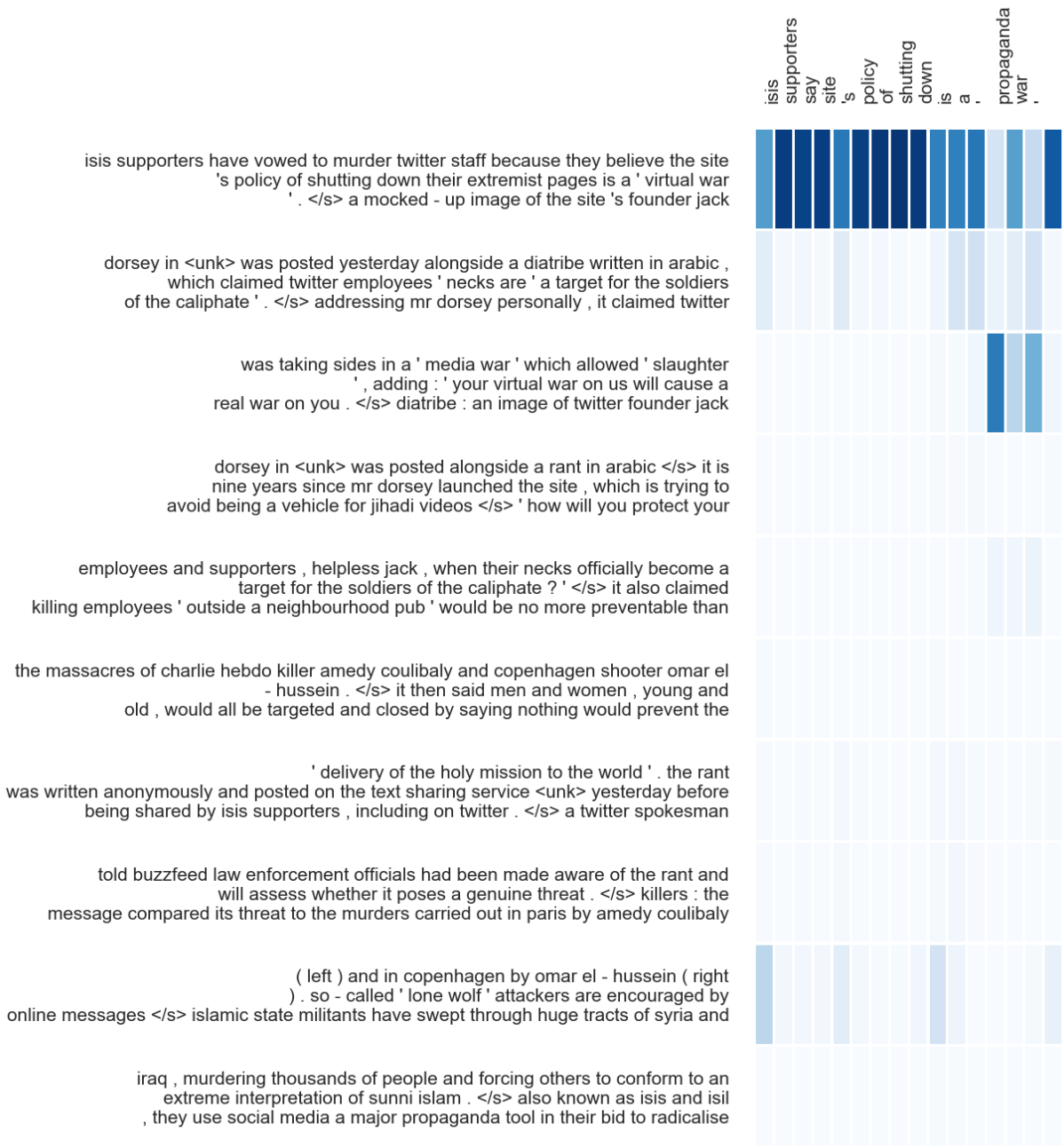



*Figure 6.11:* MODEL 3 +MULTI2


*Figure 6.12:* MODEL 3 +MULTI2 +POS

# Chapter 7

# Discussion

As shown by the results, coarse-to-fine attention generally performs worse than standard sequence-to-sequence attention.

We first discuss the weakness of the soft attention model, i.e. MODEL 2. As we noted, we may be obtaining poor results because of an optimization problem. The low entropy of MODEL 0 attention, as well as the attention visualization, both show that the model mainly focuses on one sentence at a time when generating the output. Thus, in theory there exists an attention distribution for the coarse-to-fine models to replicate, but the model is failing to learn it.

Assuming we can learn the attention distribution, another possible cause for the worse performance of coarse-to-fine models is lower modeling power. Because we run the fine-grained LSTM across each sentence separately, it may not capture the full context as well as running the LSTM across the entire document as in MODEL 0. However, we found in additional experiments that even if we ran the LSTM over the full document for the fine-grained encoding, there was not a significant change in performance. This does not conclusively rule out this possibility, but only shows that the performance bottleneck is in the attention.

Another modeling issue is in the sentence representations. While bag-of-words and convolutional models are effective for simple tasks, we would be interested to see if more sophisticated representations can allow us to obtain better attention. There is some research on developing general sentence representations for transfer learning (Bowman et al., 2016), and we leave this area to future work.

Despite the worse performance of MODEL 2 and its hard counterpart MODEL 3, there are interesting aspects of the hard attention model. First, we see that the model does

indeed learn to attend to a small subset of sentences. This suggests that the model may in fact be useful in large-scale conditional computation frameworks; however, this will definitely depend on how the performance scales as we increase the number of sentences.

Second, sampling multiple times with $k_{mul} > 1$ seems to do well, and only incurs a constant cost in computational complexity. By relaxing the constraint that we attend to a single sentence, we can build models with similar complexity but better training guarantees.

Third, we see that pretraining the hard attention model with soft attention helps performance. While pretraining with soft attention on the task at hand defeats the purpose of conditional computation, we can imagine starting with a pretrained LSTM on the encoder side from another task to possibly achieve the same effect. In addition, we see that the fixed bag-of-words coarse attention performs almost as well as the higher powered convolutional model — because the word embeddings were fixed in this case, we conclude that lower variance in the training process will lead to better performance. Transfer learning of a pretrained LSTM may partially solve this problem.

We conclude that while reinforcement learning has promise in building large-scale models with conditional computation, the noisiness of training is a difficult barrier to overcome. There are possible directions in reducing the burden of RL training (e.g., the aforementioned pretraining), but further research is needed.

## 7.1 Future Work

We propose the following as concrete future work.

We would like to explore alternative methods of sentence representation to improve the coarse attention. These methods would need to be able to encode a sentence faster than the fine-grained LSTM encoder, and as a bonus, should be trainable in an unsupervised way on an auxiliary text corpus (as in word2vec).

We may reconsider our approach to conditional computation. As we have seen, reinforcement learning is a very noisy training method, and has seen limited success in NLP so far. Alternative methods we can consider include approximate nearest neighbors (ANN) techniques (Rae et al., 2016) and the key-value memory networks of Miller et al. (2016). The latter approach is similar to our coarse-to-fine attention, as it selects paragraphs of text based on word overlap with a query. While they do not train this selection mechanism, this may in fact be an advantage due to The former approachANN

techniques would likely be more involved.

We also would like to scale up the summarization task. In our experiments, we use truncated news articles, and the sentence selection problem is relatively easy (choosing 1 sentence out of 10 in our experiments). We would be interested to see if our techniques can be applied to larger documents, such as Wikipedia articles or scientific papers, without sacrificing too much in terms of performance. We hypothesize that this will probably not be possible without somehow pretraining the encoder. Unfortunately, another bottleneck we will likely run into is GPU memory. Experiments of scale will require a large amount of computational resources, but also gives us the opportunity to fully test the complexity advantage of our coarse-to-fine method.

We would also be interested to see if our model can be applied to other text processing tasks such as question-answering (QA). QA is a more clearly posed problem than summarization — answering a question correctly is much less ambiguous than producing an accurate summary, and the metrics tend to be more informative as well. Classical QA tasks have relied on the use of Knowledge Bases such as Freebase (Bollacker et al., 2008), but there has been work in using free-form text such as Wikipedia as an information source (Miller et al., 2016). While our model may have been designed for summarization, the coarse-to-fine attention mechanism is general and should be applicable to these problems.

# Chapter 8

# Conclusion

In this thesis, we explored the problem of document summarization from a deep learning perspective. While summarization has a rich history of methods, very few are truly abstractive — most require some form of extraction from the text, perhaps followed by minor transformations and paraphrasing.

Deep learning, on the other hand, takes a truly generative approach. The sequence-to-sequence with attention model of NLP can read a document and produce a summary, all while acting on only vector space representations of the text.

While seq2seq models are state-of-the-art on several tasks including short text summarization, the computational cost is nontrivial, especially once we consider problems of scale such as document summarization. We saw that one of the biggest bottlenecks of seq2seq methods is the attention mechanism. The standard seq2seq model requires that we process the full input sequentially, and then repeatedly attend to all of it throughout the output generation process. Such a method is highly inefficient, especially since only a small fraction of the source is usually necessary for producing the output.

Therefore, we develop a new coarse-to-fine attention architecture to reduce the computational complexity. By drawing on ideas of conditional computation from deep learning, we attend to subsets of the document at a time using coarse representations of the text. By employing hard attention at the coarse level to reduce the input size, we dramatically reduce the cost of computing the more expensive fine-grained representations.

Due to its nondifferentiability, we train our model using methods from reinforcement learning. We explore the stochastic computation graph framework to rigorously understand the training algorithm, and note that the algorithm fits in naturally with existing deep learning frameworks.

We experiment with our coarse-to-fine model on the CNN/Dailymail dataset. We find that it fails to beat the standard sequence-to-sequence model on metrics, but has the desired property of sharp attention on a small subset of the source. We therefore are hopeful that such an attention mechanism can scale up existing deep learning models to larger inputs.

While we have found promising results using coarse-to-fine attention, further research is necessary to fully understand its role. In future work, we propose to scale up the task to better understand the computational gains, and also to apply the model to other NLP tasks such as question-answering.

Our method is just one of many in the recent flood of deep learning ideas in NLP. Because the field is moving so incredibly fast, we are excited to integrate our model with the latest work and slowly push machines towards human-level natural language understanding.

# Bibliography


Ba, Jimmy, Mnih, Volodymyr, and Kavukcuoglu, Koray. Multiple Object Recognition with Visual Attention. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

Bahdanau, Dzmitry, Cho, Kyunghyun, and Bengio, Yoshua. Neural Machine Translation By Jointly Learning To Align and Translate. *Iclr 2015*, pp. 1–15, 2014.

Bengio, Yoshua, Ducharme, Réjean, Vincent, Pascal, and Janvin, Christian. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

Bengio, Yoshua, Léonard, Nicholas, and Courville, Aaron C. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. *CoRR*, abs/1308.3, 2013.

Bollacker, Kurt, Evans, Colin, Paritosh, Praveen, Sturge, Tim, and Taylor, Jamie. Freebase: a collaboratively created graph database for structuring human knowledge. *SIGMOD 08 Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pp. 1247–1250, 2008.

Bowman, Samuel R., Vilnis, Luke, Vinyals, Oriol, Dai, Andrew M., Jozefowicz, Rafal, and Bengio, Samy. Generating Sentences from a Continuous Space. *Iclr*, pp. 1–13, 2016.

Brown, Peter F, Della Pietra, Vincent J, Della Pietra, Stephen A, and Mercer, Robert L. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

Cao, Ziqiang, Wei, Furu, Li, Sujian, Li, Wenjie, Zhou, Ming, and Wang, Houfeng. Learning Summary Prior Representation for Extractive Summarization. *Proceedings ACL 2015*, pp. 829–833, 2015.

Carbonell, J and Goldstein, J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 335–336, 1998.

Chen, Danqi, Bolton, Jason, and Manning, Christopher D. A Thorough Examination of the CNN / Daily Mail Reading Comprehension Task. *Acl 2016*, pp. 2358–2367, 2016.

Cheng, Jianpeng and Lapata, Mirella. Neural Summarization by Extracting Sentences and Words. *Arxiv*, pp. 484–494, 2016.

Cohn, Trevor and Lapata, Mirella. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pp. 137–144. Association for Computational Linguistics, 2008.

Collobert, Ronan, Kavukcuoglu, Koray, and Farabet, Clément. Torch7: A matlab-like environment for machine learning. *BigLearn, NIPS Workshop*, pp. 1–6, 2011a.

Collobert, Ronan, Weston, Jason, Bottou, Léon, Karlen, Michael, Kavukcuoglu, Koray, and Kuksa, Pavel. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011b.

Duchi, John, Hazan, Elad, and Singer, Yoram. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Durrett, Greg, Berg-Kirkpatrick, Taylor, and Klein, Dan. Learning-Based Single-Document Summarization with Compression and Anaphoricity Constraints. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1998–2008, 2016.

Farabet, Clement, Couprie, Camille, Najman, Laurent, and LeCun, Yann. Learning Hierarchical Features for Scence Labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–15, 2013.

Goldberg, Yoav. A primer on neural network models for natural language processing. *arXiv preprint arXiv:1510.00726*, 2015.

Hermann, KM, Kocisky, T, and Grefenstette, E. Teaching machines to read and comprehend. *Advances in Neural Information Processing Systems*, pp. 1–9, 2015.

Hinton, Geoffrey, Deng, Li, Yu, Dong, Dahl, George E., Mohamed, Abdel-rahman, Jaitly, Navdeep, Senior, Andrew, Vanhoucke, Vincent, Nguyen, Patrick, Sainath, Tara N., and Others. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jurafsky, D and Martin, J. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall series in artificial intelligence, 2nd ed edition, 2009. ISBN 0130950696.

Kim, Yoon. Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1746–1751, 2014.

Kingma, Diederik P. and Ba, Jimmy Lei. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations 2015*, pp. 1–15, 2015.

Knight, Kevin and Marcu, Daniel. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, 2002.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pp. 1–9, 2012.

Kupiec, Julian, Pedersen, Jan, and Chen, Francine. A Trainable Document Summarizer. *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 68–73, 1995.

LeCun, Y and Bengio, Y. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(April 2016):255–258, 1995.

Li, Jiwei, Luong, Minh-Thang, and Jurafsky, Dan. A Hierarchical Neural Autoencoder for Paragraphs and Documents. *CoRR*, abs/1506.0, 2015.

Li, Jiwei, Galley, Michel, Brockett, Chris, Gao, Jianfeng, and Dolan, Bill. A Persona-Based Neural Conversation Model. *arXiv preprint arXiv:1603.06155*, 2016.

Lin, Chin-Yew. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.

Luhn, Hans Peter. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

Luong, Minh-Thang, Pham, Hieu, and Manning, Christopher D. Effective Approaches to Attention-based Neural Machine Translation. *Emnlp*, (September):11, 2015.

Manning, Christopher D. Computational Linguistics and Deep Learning, 2016.

Martins, André F. T. and Astudillo, Ramón Fernandez. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1614–1623, 2016.

Mei, Hongyuan, Bansal, Mohit, and Walter, Matthew R. What to talk about and how? Selective Generation using LSTMs with Coarse-to-Fine Alignment. *Proceedings of NAACL-HLT*, pp. 1–11, 2016.

Mikolov, Tomas and Dean, Jeffrey. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 2013.

Miller, Alexander, Fisch, Adam, Dodge, Jesse, Karimi, Amir-Hossein, Bordes, Antoine, and Weston, Jason. Key-Value Memory Networks for Directly Reading Documents. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP-16)*, abs/1606.0:1400–1409, 2016.

Mnih, Volodymyr, Heess, Nicolas, Graves, Alex, and koray Kavukcuoglu. Recurrent models of visual attention. *Advances in Neural Information Processing Systems*, pp. 2204—-2212, 2014.

Nallapati, Ramesh, Zhou, Bowen, dos Santos, Cicero Nogueira, Gulcehre, Caglar, and Xiang, Bing. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. *Proceedings of CoNLL*, abs/1602.0:280–290, 2016.

Nenkova, Ani and McKeown, Kathleen. Automatic Summarization. *Foundations and Trends® in Information Retrieval*, 5(3):235–422, 2011.

Over, Paul, Dang, Hoa, and Harman, Donna. DUC in context. *Information Processing & Management*, 43(6):1506–1520, 2007.

Papineni, Kishore, Roukos, Salim, Ward, Todd, and Zhu, Wei-jing. BLEU: a Method for Automatic Evaluation of Machine Translation. *Computational Linguistics*, (July):311–318, 2002.

Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

Rae, Jack, Hunt, Jonathan J, Danihelka, Ivo, Harley, Timothy, Senior, Andrew W, Wayne, Gregory, Graves, Alex, and Lillicrap, Tim. Scaling Memory-Augmented Neural Networks with Sparse Reads and Writes. In Lee, D D, Sugiyama, M, Luxburg, U V, Guyon, I, and Garnett, R (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3621–3629. Curran Associates, Inc., 2016.

Ranzato, Marc'Aurelio, Chopra, Sumit, Auli, Michael, and Zaremba, Wojciech. Sequence Level Training with Recurrent Neural Networks. *CoRR*, abs/1511.0:1–15, 2015.

Rosenblatt, F. A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

Rush, Alexander M, Chopra, Sumit, and Weston, Jason. A Neural Attention Model for Abstractive Sentence Summarization. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

Schulman, John, Heess, Nicolas, Weber, Theophane, and Abbeel, Pieter. Gradient Estimation Using Stochastic Computation Graphs. *NIPS*, pp. 1–13, 2015.

Shannon, Claude E. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(July 1948):379–423, 1948.

Shazeer, Noam, Mirhoseini, Azalia, Maziarz, Krzysztof, Davis, Andy, Le, Quoc, Hinton, Geoffrey, and Dean, Jeff. Outrageously Large Neural Networks: the Sparsely-Gated Mixture-of-Experts Layer. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Silver, David, Huang, Aja, Maddison, Chris J., Guez, Arthur, Sifre, Laurent, van den Driessche, George, Schrittwieser, Julian, Antonoglou, Ioannis, Panneershelvam, Veda, Lanctot, Marc, Dieleman, Sander, Grewe, Dominik, Nham, John, Kalchbrenner, Nal, Sutskever, Ilya, Lillicrap, Timothy, Leach, Madeleine, Kavukcuoglu, Koray, Graepel, Thore, and Hassabis, Demis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

Sutskever, Ilya, Vinyals, Oriol, and Le, Quoc V. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Toutanova, Kristina, Tran, Ke M, and Amershi, Saleema. A Dataset and Evaluation Metrics for Abstractive Compression of Sentences and Short Paragraphs. In *EMNLP*, nov 2016.

Weaver, Lex and Tao, Nigel. The optimal reward baseline for gradient-based reinforcement learning. *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pp. 538–545, 2001.

Williams, Ronald J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Wu, Yonghui, Schuster, Mike, Chen, Zhifeng, Le, Quoc V., Norouzi, Mohammad, Macherey, Wolfgang, Krikun, Maxim, Cao, Yuan, Gao, Qin, Macherey, Klaus, Klingner, Jeff, Shah, Apurva, Johnson, Melvin, Liu, Xiaobing, Kaiser, Łukasz, Gouws, Stephan, Kato, Yoshikiyo, Kudo, Taku, Kazawa, Hideto, Stevens, Keith, Kurian, George, Patil, Nishant, Wang, Wei, Young, Cliff, Smith, Jason, Riesa, Jason, Rudnick, Alex, Vinyals, Oriol, Corrado, Greg, Hughes, Macduff, and Dean, Jeffrey. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *ArXiv e-prints*, pp. 1–23, 2016.

Xu, Kelvin, Ba, Jimmy, Kiros, Ryan, Cho, Kyunghyun, Courville, Aaron C, Salakhutdinov, Ruslan, Zemel, Richard S, and Bengio, Yoshua. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *ICML*, 14:77—-81, 2015.

Zajic, David, Dorr, Bonnie, and Schwartz, Richard. Bbn/umd at duc-2004: Topiary. In *Proceedings of the HLT-NAACL 2004 Document Understanding Workshop, Boston*, pp. 112–119, 2004.

Zeiler, Matthew D. and Fergus, Rob. Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8689 LNCS(PART 1):818–833, 2014.

# Appendix A

# Full Source Documents

**Document 1, Figure 5.1** ( cnn ) the man suspected of killing a deputy u.s. marshal at a motel in baton rouge , louisiana , has died , brittany stewart in the east baton rouge coroner 's office said wednesday . </s> the cause of death is pending autopsy , she said . </s> jamie croom , ## , was wounded in a shootout with deputy u.s. marshal josie wells . </s> it can be one of the most dangerous tasks for a law enforcement officer : serving an arrest warrant to a fugitive murder suspect . when wells tried to do that tuesday , he lost his life . </s> " wells was part of a team executing arrest warrants on a fugitive wanted for double homicide in baton rouge , " the u.s. marshals service said in a news release . " the team engaged in gunfire with the fugitive and wells was shot . he was immediately transported to lane regional medical center in zachary , louisiana , where he died . " </s> wells , ## , was trying to arrest croom , who is suspected in the deaths of a brother and sister in new roads , louisiana , cnn affiliate wafb said . </s> " our deputies and law enforcement partners face untold dangers every day in the pursuit of justice in cities nationwide , " said u.s. marshals service director stacia a. hylton said in a news release . " the fugitive who killed deputy wells was extremely dangerous , wanted for double homicide and intentionally evaded justice . ... when a public servant dies in the line of duty , it is an immeasurable tragedy felt by all . our thoughts and prayers are with deputy wells ' family , friends and colleagues . " </s> officials would not elaborate on exactly what happened , but the advocate newspaper in baton rouge said there was a shootout . authorities said croom shot and killed wells at a baton rouge motel . </s> wells , ## , was based in mississippi but was on temporary assignment in the baton rouge area , the sun herald newspaper said . he was married and came from a law enforcement family . </s> despite the risks , wells loved his job . </s> " it was his passion , " longtime friend

alex mcgee told the paper . " i tipped my hat to him because he knew the dangers and wanted to do the job anyway . " </s> croom , the suspect , was taken to a hospital after he was wounded , wafb said . </s> he was wanted in the shooting deaths of the two siblings in february and was also on probation for firearms charges . </s> that double homicide stemmed from a feud over a loan made to one of the victim 's relatives as well as an alleged break - in at the suspect 's grandmother 's house , croom 's older sister latonia croom duncan told cnn . </s> duncan said the family reported threats and a shooting at her grandmother 's house to police , but said there was never any follow - up . </s> she said her brother called her the night of the homicides , which took place at a nightclub . </s> " he called me and said he loved me and that he 'd be gone , " duncan said . " he said he was n't going back to jail . before he 'd go back to jail , he said , he 'd rather be dead . " </s> cnn 's devon m. sayers , john newsome , sam stringer and eliott c. mclaughlin contributed to this report . </s>

**Document 2, Figure 5.1** ( cnn ) there have been a few times in my career when i 've been thoroughly disappointed – even disgusted – with my fellow women in the workplace . </s> no , i certainly do n't expect all my female colleagues to go out of their way for me and sing " kumbaya " together in the office , but i 'm always stunned when a woman who could have been helpful to me was n't , when a woman who could have been a mentor chose not to be , when a woman tried to hurt me because of her own fear , anxiety or what have you . </s> i 'd love to say more about each of the women i 've met along the way who fit those descriptions , but my point is not to single anyone out . my goal is to ask the question , " why ? " </s> obviously , not all women are like this and there are plenty of men guilty of the same behavior , but why do so many women try to tear each other down instead of lift each other up ? </s> i figured this would be a perfect question for sophia nelson , author of a new self - help book for women called " the woman code , " and she did n't disappoint . </s> unlocking ' the woman code ' : # tips to know your value </s> " from the time we 're little girls , we 're taught to compete , " said nelson during a recent conversation at cnn . " i need to be prettier , taller , smarter , my hair needs to be straighter , curlier , whatever it is . i need to get the better looking guy . i need to always be better than because we 're taught to come from a place of lack as women . " </s> the way nelson , an award - winning author and journalist , radio and television personality and motivational speaker , sees it , we women need to start operating like the boys . </s> men " operate from a sense of , there 's this whole pie , and i want my piece

, and i do n’t care if he gets his piece , and maybe we even have to work together to start that business , start that company , ” said nelson . </s> of course , it ’s easier for a man not to worry ” if he gets his piece ” since there are plenty of pieces of pie available for men in terms of management positions in corporate america , but that is n’t the case for women . today , just # % of s&p ### chief executives are women and only ## % of the top five senior leadership positions at those companies are held by women , according to a cnn money analysis . </s> sheryl sandberg teams up with nba to get men to # leanin </s> decades ago , the situation was even worse . when i was just starting my television news career in #### , women who were in their ##s and were in high - level positions were the only women in a position of influence . naturally , many of them often viewed other women as threats who could take their job . </s> ” because they did n’t think there could be ten of them , they only thought there could be one of them , ” said nelson . ” fast forward ## years later . now there ... are a number of women partners at big firms , a number of women in congress . i could keep going on and on so ... there is a place for more of us . ” </s> which means we can lift as we climb , we can help our younger sisters and even our cohorts while still moving up and on in our careers , says nelson . </s> ” how exactly do we do that ? ” i had to ask . nelson came armed with five tips on how women can work with as opposed to against each other . </s> first , nelson says be mindful of the people you surround yourself with and careful about ” who ’s in your row . ” </s> ” if you hear another woman say , and i ’ve heard this , ’ i do n’t do women friends , i do n’t have women friends , ’ believe her and leave her alone . i mean that , listen to me now , ” said nelson . </s> there are too many women who believe in the sisterhood of women , so do n’t invest any time , if possible , with people who do n’t , she says . </s> if you are in a meeting and you have a great idea , do n’t feel like you have to hoard it to yourself , said nelson . ” collaborate , share , collaborate . ... so you lift other women as you climb by collaborating versus competing . ” </s> competition is healthy and we can compete , but we ought to take a page from our male colleagues ’ playbook , she said . ” the guys collaborate better than we do because they operate from a place of ’ i want the dollars . i want to win the contract . i want to get the business . ’ we have to get in that same mindset . ” </s> we ’re all busy but we ’ve got to slow down and mentor , said nelson . </s> ” we have to build a bench , ” she said . ” men do this well again . you ’ve seen it in corporate , i ’ve seen it . the guys go out and golf . they do things together and they ’re building up the next young man leader . whatever field we ’re in ... we ’re less likely to do it because we ’re busy . we ’ve got to mentor . ” </s> when you lift other

women as you climb , said nelson , you realize it 's reciprocal . " it 's not all about you . " </s> we women win when more women are in executive roles in organizations , i added . </s> " the right women , " said nelson . " i want to caveat that . and again , i do n't mean to be mean or catty but ... i know a lot of women in power positions that do n't help other women but there are a lot of women in power positions that do . " </s> this is a tough one for us , says nelson . we need to be willing to say to another woman that we did n't like something she did or said and do it in a respectful and private way where we are still building her up , not pulling her down . </s> " do n't go tell ## of your friends not to like her . you 'd be amazed at how silly we can be . we 're still in kindergarten some of us , " said nelson . </s> " gossip is still one of the most rampant , nasty things we do as women to each other . and it hurts . it really damages women . " </s> why do you think women too often tear each other down instead of help each other in the workplace ? share your thoughts with kelly wallace on twitter @ kellywallacetv or cnn living on facebook . </s>

**Document 3, Figure 5.1** much of the start of the world 's most famous sled dog race is covered in barren gravel , forcing iditarod organizers to move the start further north where there is snow and ice . </s> a weather pattern that buried the eastern u.s. in snow has left alaska fairly warm and relatively snow - free this winter , especially south of the alaska range . </s> ' if i have one more person say to me to move the iditarod to boston , i 'm going to shake my head , ' said race director mark nordman . </s> scroll down for video in this photo taken on thursday , there are bare patches of grass and mud on sled dog trails in anchorage , alaska which is unsuitable for the iditarod </s> the iditarod trail sled dog race starts saturday with a ceremonial run through anchorage . but the official start two days later has been moved ### miles ( ### kilometers ) north , over the alaska range , to fairbanks to avoid the area that left many mushers bruised and bloodied last year . iditarod officials said the conditions are worse this year . </s> the race 's chief executive officer , stan hooley , called the conditions ' pretty miserable . ' and last year was no picnic . </s> one musher last year was taken out by a rescue helicopter after making it through the dalzell gorge only to hit his head on a tree stump in the farewell burn . knocked unconscious for at least an hour , scott janssen got back on the trail after waking up . but shortly after , he broke his ankle while walking on ice trying to corral a loose dog . </s> mush ! : aliy zirkle 's team runs across willow lake during the iditarod trail sled dog race in willow lake , alaska last year when the terrain was snowy enough '

as an outdoorsman , to have to be rescued from the trail is n't a wonderful thing , ' janssen said . </s> this year 's race will feature ## mushers , including six former champions and ## rookies . the winner is expected in nome in about ## days . </s> alaskans can thank the jet stream , which has been delivering warm air from the pacific , said dave snider , a meteorologist with the national weather service in anchorage . </s> ' that position of the jet has been pretty stagnant , or at least in the general same position for a long period of time . while that 's allowing a lot of cold air to flow out of the arctic into the midwest and the eastern seaboard , we 're locked into the warmer part of that pattern , and we 've continued to see those warm pushes for a fairly long period over the winter , ' he said . </s> anchorage gets about ## inches ( ### centimeters ) of snow in a normal year ; this year only about ## inches ( ## centimeters ) have fallen . </s> the new route , which puts mushers on river ice for about ### miles , could level the playing field . </s> there are bare patches of grass and mud on sled dog trails in anchorage , alaska making it not ideal to hold the world famous dog sled race </s> ' nobody has a plan , ' nordman said . ' you 're not going to be stopping and putting your snow hook into the same tree you had the last ## years . it 's a whole new ballgame . ' </s> brent sass of eureka , alaska , is running his third iditarod , and is coming off a win in last month 's #,### - mile ( #,### - kilometer ) yukon quest international sled dog race . </s> ' it does n't hurt a guy like me who has only run the race a couple of times , ' he said of the route change . ' for the guys that have run the race ## times , it 's not just the normal routine so it might throw them off a little bit . ' </s> among the veterans in this year 's race is the defending champion , dallas seavey , and the #### bizarre finish will be remembered as much as the poor trail conditions . </s> stan hooley , the chief executive officer of the iditarod trail sled dog race , is silhouetted as video of a musher wiping out in the #### race plays behind him during a meeting in anchorage , alaska , ahead of the #### race </s> a sudden blizzard blew four - time champion and race leader jeff king out of the race when he was about ## miles ( ## kilometers ) from the finish line of the nearly #,### - mile race . </s> then aliy zirkle , who was solidly in second place , waited out the storm at the last checkpoint , ## miles ( ## kilometers ) from nome , for two hours , ## minutes . she got back on the trail when seavey blew through the checkpoint , but lost the race by two minutes , ## seconds . it was her third straight runner - up finish with no wins . </s> the route change eliminates the mountainous terrain and treacherous gorge , but it could present mushers with a whole new set of problems with a flat trail on unpredictable river ice . plus , because it 's an entirely new route , mushers say they ca n't rely much

on information , even something as simple as the mileage between village checkpoints , provided by iditarod officials . </s> by removing the alaska range , mushers may assume it will be a very fast race , seavey said . </s> ' just because it 's a flat trail does not mean your dogs can all of a sudden do ## times what they 've been able to do in the past , ' said seavey , a two - time champion . ' i feel that is a trap that will catch a lot of people . ' </s> ' in the end , this race will not be won on tricks or gimmicks . it will be won on good dogmanship , ' he said . </s> too warm : in this photo taken on wednesday , stan hooley , the chief executive officer of the iditarod trail sled dog race , speaks in anchorage , alaska , ahead of the #### race </s>

**Document of Figure 6.2** jasmine coleman , ## , has been found safe and well some ## miles from her home a ## - year - old girl who went missing from her family home at #am amid fears she was driven away by an ' older man ' has been found safe and well . jasmine coleman was reported as missing this morning after disappearing from her home in lancing , west sussex . the child was found this afternoon following a police appeal some ##miles away in croydon , south east london . police feared she may have been driven to london by an older man when they launched an appeal for information this morning . the schoolgirl had not been seen since ##.##pm on friday night . sussex police said she may have been talking with someone on facetime before disappearing at around #am . the force launched a public appeal for information on her whereabouts on saturday morning . in it , she was described as fair with long , blonde hair and as having possibly been wearing black riding trousers and a polo shirt or a paisley pattern dress . on saturday afternoon the force confirmed she had been found safe and well in croydon but could not confirm the circumstances under which police located her . no one has been arrested in connection with her disappearance . shortly after #pm a spokesman said : ' jasmine coleman , the ## - year - old girl from lancing reported missing from home during the early hours of saturday ( ## march ) was found safe and well later the same day in croydon . ' police would like to thank the media and members of the public for help given during efforts to trace her . ' west sussex police ( headquarters pictured above ) confirmed the child was found on saturday afternoon </s>

**Document of Figure 6.3** isis supporters have vowed to murder twitter staff because they believe the site 's policy of shutting down their extremist pages is a ' virtual war ' . </s> a mocked - up image of the site 's founder jack dorsey in cross-hairs was posted

yesterday alongside a diatribe written in arabic , which claimed twitter employees ' necks are ' a target for the soldiers of the caliphate ' . </s> addressing mr dorsey personally , it claimed twitter was taking sides in a ' media war ' which allowed ' slaughter ' , adding : ' your virtual war on us will cause a real war on you . </s> diatribe : an image of twitter founder jack dorsey in cross-hairs was posted alongside a rant in arabic </s> it is nine years since mr dorsey launched the site , which is trying to avoid being a vehicle for jihadi videos </s> ' how will you protect your employees and supporters , helpless jack , when their necks officially become a target for the soldiers of the caliphate ? ' </s> it also claimed killing employees ' outside a neighbourhood pub ' would be no more preventable than the massacres of charlie hebdo killer amedy coulibaly and copenhagen shooter omar el - hussein . </s> it then said men and women , young and old , would all be targeted and closed by saying nothing would prevent the ' delivery of the holy mission to the world ' . the rant was written anonymously and posted on the text sharing service pastebin yesterday before being shared by isis supporters , including on twitter . </s> a twitter spokesman told buzzfeed law enforcement officials had been made aware of the rant and will assess whether it poses a genuine threat . </s> killers : the message compared its threat to the murders carried out in paris by amedy coulibaly ( left ) and in copenhagen by omar el - hussein ( right ) . so - called ' lone wolf ' attackers are encouraged by online messages </s> islamic state militants have swept through huge tracts of syria and iraq , murdering thousands of people and forcing others to conform to an extreme interpretation of sunni islam . </s> also known as isis and isil , they use social media a major propaganda tool in their bid to radicalise young muslims and persuade them to fight in iraq and syria . </s> last year home secretary theresa may said ### brits had gone to fight in the two countries , with cases emerging since then of schoolgirls fleeing to become jihadi brides . the terror group has posted gruesome videos and images of murder , including beheadings , on mainstream sites such as twitter and youtube alongside the hidden parts of the web . </s> both sites have a policy of taking down extremist content and blocking users who upload it , but usually by the time an account is shut down the content has spread halfway around the world . </s>

# Appendix B

# Attention Visualizations

Here we show some more attention heatmaps from the models, including for the summaries from Figures 6.2 and 6.4.