How to train your self-supervised NLP model: Investigating pre-training objectives, data, and scale

Mandar Joshi

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2022

Reading Committee:
Luke S. Zettlemoyer, Co-Chair
Daniel S. Weld, Co-Chair
Yejin Choi

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

How to train your self-supervised NLP model: Investigating pre-training objectives, data, and scale

Mandar Joshi

Co-chairs of the Supervisory Committee:

Professor Luke S. Zettlemoyer
Computer Science and Engineering

Professor Daniel S. Weld
Computer Science and Engineering

A robust language processing machine should be able to encode linguistic and factual knowledge across a wide variety of domains, languages, and even modalities. The paradigm of pre-training self-supervised models on large text corpora has driven much of recent progress towards this goal. In spite of this large scale pre-training, the best performing models have to be further fine-tuned on downstream tasks – often containing hundreds of thousands of examples – to achieve state of the art performance. The aim of this thesis is twofold: (a) to design efficient scalable pre-training methods which capture different kinds of linguistic and world knowledge, and (b) to enable better downstream performance with fewer human-labeled examples.

The first part of the thesis focuses on self-supervised objectives for reasoning about relationships between pairs of words. In NLI, for example, given the premise "*golf is prohibitively expensive*", inferring that the hypothesis "*golf is a cheap pastime*" is a contradiction requires one to know that *expensive* and *cheap* are antonyms. We show that with the right kind of self-supervised objectives, such knowledge learned with word *pair* vectors (`pair2vec`) directly from text without using curated knowledge bases and ontologies.

The second part of the thesis seeks to build models which encode knowledge beyond word pair relations into model parameters. We present SpanBERT, a scalable pre-training method that is designed to better

represent and predict spans of text. Span-based pre-training objectives seek to efficiently encode a wider variety of knowledge, and improve the state of the art for a range of NLP tasks.

The third part of the thesis focuses integrating dynamically retrieved textual knowledge. Specifically, even large scale representations are not able to preserve all factual knowledge they have "read" during pre-training due to the long tail of entity and event-specific information. We show that training models to integrate background knowledge during pre-training is especially useful for downstream tasks which require reasoning over this long tail.

The last part of the thesis targets a major weakness of self-supervised models – while such models requires no explicit human supervision during pre-training, they still need lots of human-labeled downstream task data. We seek to remedy this by mining input-output pairs (and thus obtaining direct task-level supervision) from corpora using supervision from very few labeled examples.

Overall, this thesis presents a range of ideas required for effective pre-training and fine-tuning – (a) self-supervised objectives, (b) model scale, and (c) new types of data. As we will show in the following chapters, *self-supervised objectives* could have a large influence on the form of knowledge that is acquired during pre-training. Moreover, efficient objectives directly enable *model scale* both in terms of data and parameters. Finally, the training *data* and the kind of supervision derived from it itself dictates how well a model can learn different kinds of downstream tasks.

Acknowledgements

I decided I was going to write this section first. It seemed like the right thing to do before I take up the task of summarizing my six-plus years of work at the University of Washington.

First and foremost, I'd like to thank my advisers Luke Zettlemoyer and Dan Weld. I could go on and on about their insightfulness and brilliance, but to me their defining traits as advisers were their personal attention and kindness. Despite leading groups at multiple institutions, both Luke and Dan always made themselves available to discuss any and every aspect my research and professional life. They taught me to be easy on myself when the chips were down. Without their generosity, my PhD would have been immensely stressful.

I have been lucky to have had mentors who were willing to climb down into the trenches with me. Some of the most fun parts of my PhD were spent working with Omer Levy who strongly influenced my taste and approach in research problems. I've missed going on swear laden rants about papers since Omer moved out of Seattle. From helping me debug Pytorch code to simplify some of my overtly complicated ideas, Mike Lewis has been the voice of reason (and sarcasm) particularly during the last two years of my PhD. I had a wonderful internship at Google thanks to Kristina Toutanova, my host. Our discussions during the internship had an influence on some of my later research directions. My thesis committee members Yejin Choi, Jianfeng Gao, and Gina-Anne Levow were instrumental in helping me put together my thesis.

One of the most fun parts of my grad school experience was being able to collaborate some wonderful people: Victoria Lin, Matt Gardner, Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Ves Stoyanov, Yi Luan, Bhargavi Paranjape, John Thickstun, Hannaneh Hajishirzi, Arie Cattan, Alon Eirew, Gabriel Stanovsky, Ido Dagan, Terra Blevins, Weijia Shi, Armen Aghajanyan, Dmytro Okhonko, Hu Xu, Gargi Ghosh, Bernie Huang and Candace Ross. I also want to especially thank Eunsol Choi, Danqi Chen, and

Kenton Lee who, as friends and co-authors, brainstormed research ideas with me when I them wanted to and looked at my ugly code when I needed them to.

A trick to completing a PhD is to stay sane and not feel lost while wandering. I am immensely thankful to fellow wanderers – Gagan Bansal, Srini Iyer, Bhargavi Paranjape, Sachin Mehta, Elizabeth Clark, Julian Michael, Kelvin Luu, Sameer Dawande, Guna Prasad, Maaz Ahmed, Jonathan Bragg, and Danielle Bragg. You were my party on good days and my shoulder to cry on during the bad ones. Those lunch trips to the Ave, the scavenging for free food, and the countless hours spent in the espresso room will always be remembered with fondness.

At several junctures, especially towards the start and end of my PhD, I received help – research feedback, introduction to people during my job search, words of kindness, and so much more – from my colleagues: Ari Holtzman, Sewon Min, Victor Zhong, Dan Fried, Tim Dettmers, Suchin Gururangan, Margaret Li, Luheng He, Mark Yatskar, Nick Fitzgerald, Swabha Swayamdipta, Sameer Singh, Jesse Thomason, and Spandana Gella. My PhD experience was so much smoother because I could always depend on the CSE staff in the event of hiccups. I owe a lot of gratitude to Elise deGoede Dorough, Elle Brown, and Joe Eckert for making sure I was able to navigate rules around internships and graduation. I also want to thank Chiemi Yamaoka-Vismale for taking care of my travel and equipment related issues through my PhD.

While my support system in Seattle kept me going, it's the folks back home in India who were my inspiration all along. There are several people in my life who were instrumental in putting me into grad school. My childhood friend Suhrid Deshmukh and my math teacher Anil Kayande planted that seed, and my research mentors Shiwali Mohan, Uma Sawant, and Soumen Chakrabarti nurtured it with all their patience. Throughout my life, my family has pushed me to become the best version of myself both personally and professionally. To them, I owe a world of gratitude.

DEDICATION

To my family

Contents

1 Introduction **17**

1.1 Motivation . . . 17

1.1.1 Self-supervised Models: Pre-training and Fine-tuning . . . 18

1.2 Background . . . 19

1.2.1 General Purpose Language Representations . . . 20

1.2.2 Task-specific Background Knowledge . . . 20

1.2.3 Data Augmentation . . . 21

1.3 Outline . . . 22

1.3.1 Efficient pre-training methods to capture linguistic and world knowledge . . . 22

1.3.2 Enabling better downstream performance with fewer human-labeled examples . . . 23

2 pair2vec: Compositional Word-Pair Embeddings for Cross-Sentence Inference **25**

2.1 Self-supervised Pretraining . . . 27

2.1.1 Representation . . . 27

2.1.2 Objective . . . 28

2.2 Integrating `pair2vec` into Models . . . 31

2.2.1 General Approach . . . 31

2.2.2 Question Answering . . . 32

2.2.3 Natural Language Inference . . . 33

2.3 Experiments . . . 34

2.3.1 Question Answering . . . 34

2.3.2 Natural Language Inference . . . 35
2.3.3 Ablations . . . 36
2.4 Analysis . . . 37
2.4.1 Quantitative Analysis: Word Analogies . . . 37
2.4.2 Qualitative Analysis: Slot Filling . . . 39
2.5 Related Work . . . 40
2.6 Conclusion and Future Work . . . 41

3 Improving Pre-training by Representing and Predicting Spans 43
3.1 Model . . . 45
3.1.1 Span Masking . . . 45
3.1.2 Span Boundary Objective (SBO) . . . 46
3.2 Single-Sequence Training . . . 47
3.3 Experimental Setup . . . 47
3.3.1 Tasks . . . 47
3.3.2 Implementation . . . 50
3.3.3 Baselines . . . 51
3.4 Results . . . 51
3.4.1 Per-Task Results . . . 51
3.4.2 Overall Trends . . . 53
3.5 Ablation Studies . . . 55
3.5.1 Masking Schemes . . . 55
3.5.2 Auxiliary Objectives . . . 56
3.6 Related Work . . . 56
3.7 Conclusion . . . 58

4 Representations Using Dynamically Retrieved Textual Knowledge 59
4.1 TEK-Enriched Representations . . . 61
4.1.1 TEK-Enriched Question Answering . . . 62

4.1.2 TEK-Enriched Pretraining . . . 63
4.2 Experimental Setup . . . 64
4.2.1 Baselines . . . 65
4.3 Results . . . 65
4.4 Ablation Studies . . . 68
4.5 Discussion . . . 69
4.6 Related Work . . . 71
4.7 Conclusion . . . 73

5 Few-shot Mining of Naturally Occurring Inputs and Outputs 75

5.1 Mining . . . 77
5.1.1 Coarse-grained Search . . . 77
5.1.2 Fine-grained Filtering . . . 79
5.2 Application to Tasks . . . 79
5.2.1 Reading Comprehension . . . 79
5.2.2 Summarization . . . 80
5.3 Experiments . . . 80
5.3.1 Benchmarks . . . 80
5.3.2 Baselines . . . 81
5.3.3 Main Results . . . 82
5.4 Ablations . . . 83
5.4.1 Amount of Mined Data . . . 84
5.4.2 Bi vs Crossencoder Differences . . . 84
5.4.3 Naturally Occurring vs. Model Generated Data . . . 84
5.5 Discussion . . . 84
5.5.1 XSum . . . 84
5.5.2 SQuAD . . . 86
5.6 Related Work . . . 86
5.7 Conclusion . . . 88

6 Conclusion **89**

6.1 Future Work 90

List of Figures

1.1 An article from Sportstar. 18

2.1 A typical architecture of a cross-sentence inference model (left), and how `pair2vec` is added to it (right) 30

2.2 Accuracy as a function of the interpolation between fastText and `pair2vec` 37

3.1 An illustration of SpanBERT training 44

3.2 SpanBERT masking: Sampling random span lengths from a geometric distribution 45

4.1 A TriviaQA example showing how background sentences from Wikipedia help question answering. 59

4.2 The TEK-enriched encoding scheme 61

4.3 Case study of TEK model strengths 70

5.1 Examples of mined input output pairs for summarization and reading comprehension. 76

5.2 A pipeline of coarse and fine-grained models for mining high-quality data 77

5.3 Performance (ROUGE-L) on the XSum dev set of various augmentation techniques with varying amounts of augmented data added to X_{100}. 82

5.4 Performance (F1) on the SQuAD dev set of various augmentation techniques with varying amounts of augmented data added to X_{100}. 83

List of Tables

2.1 Example word pairs with contexts. . . . 26

2.2 The bivariate and multivariate negative sampling objectives for `pair2vec`. . . . 28

2.3 `pair2vec` performance on SQuAD . . . 34

2.4 `pair2vec` performance on MultiNLI . . . 35

2.5 `pair2vec` performance on the adversarial NLI . . . 35

2.6 `pair2vec` ablations on SQuAD . . . 36

2.7 The top 10 analogy relations for which interpolating with `pair2vec` improves performance 37

2.8 Top 3 words from the entire vocabulary for example word and contexts . . . 38

3.1 SpanBERT results on SQuAD 1.1 and SQuAD 2.0 . . . 52

3.2 SpanBERT performance (F1) on MRQA . . . 52

3.3 SpanBERT performance on coreference resolution . . . 52

3.4 SpanBERT performance on TACRED . . . 53

3.5 SpanBERT peformance on GLUE . . . 53

3.6 The effect of masking schemes for BERT based models . . . 54

3.7 The effects of different auxiliary objectives for SpanBERT . . . 54

4.1 Data statistics for TriviaQA and MRQA. . . . 64

4.2 TEK Performance on TriviaQA . . . 66

4.3 TEK performance on MRQA . . . 67

4.4 Model performance using different combinations of pretraining and finetuning . . . 68

4.5 F1 on TriviaQA and MRQA for varying lengths of context and background lengths . . . 68

4.6 Performance of 12-layer TEK$_{PF}$ when used with publicly available entity linkers on TriviaQA and MRQA 69

5.1 Performance on the SQuAD dev set after training on 100 examples. 81

5.2 Performance on the XSum test set after training on 100 examples. Both BART and Us use the same training examples. * indicates that results have been taken from Fabbri et al. [2021] 81

5.3 Mined examples from the bi and crossencoders. Answer spans are indicated in *italics*. 85

Chapter 1

Introduction

1.1 Motivation

Reasoning about relationships between words, entities, sentences, and larger textual units is critical to understanding natural language. For example, consider this headline from the Sportstar magazine[1], "With ball, Tendulkar junior helps England ahead of Australia clash" in Figure 1.1. Cricket fans, the intended readers of the article, might be able to assume that the phrase *Tendulkar junior* refers to *Arjun Tendulkar*, the son of famous cricketer Sachin Tendulkar. Likewise, the phrase *with ball* has a domain-specific meaning, and refers to bowling in cricket,[2] and that *clash* refers to an upcoming match between the English and Australian national cricket teams. In other words, understanding this piece of text requires a range of linguistic and factual knowledge. A robust language processing machine should therefore be able to encode such knowledge across a wide variety of domains, languages, and even modalities.

Much of research in natural language processing (NLP) seeks to build and evaluate models on their ability to encode, represent, and use this knowledge. For example, the task of coreference resolution could test whether the model is able to correctly predict that the phrases *Tendulkar junior* and *Arjun Tendulkar* are coreferent. Other such tasks could test for parts of speech, meaning of words or relation between them, syntactic parsing etc.

A complementary view of language processing would be to not just to test for factual and linguistic

[1] https://sportstar.thehindu.com/cricket/icc-cricket-world-cup/news/world-cup-2019-england-vs-australia-arjun-tendulkar-nets-bowling-practice-lords-sachin-tendulkar/article28126936.ece

[2] `https://en.wikipedia.org/wiki/Bowling_(cricket)`

With ball, Tendulkar junior helps England ahead of Australia clash

Aiming to live up to his illustrious surname, Arjun Tendulkar ran in fast and bowled at the England batsmen on the eve of their big-ticket clash against arch-rival Australia.

PTI

LONDON 24 JUNE, 2019 19:37 IST

A Tendulkar helped England ahead of its World Cup clash against Australia – not the man who owns almost all the batting records but the boy who aspires to shine with the ball.

Aiming to live up to his illustrious surname, Arjun Tendulkar ran in fast and bowled at the England batsmen on the eve of their big-ticket clash against arch-rival Australia.

The son of batting legend Sachin Tendulkar Monday marked his run-up at the Lord's nets during England's practice session, drawing, on expected lines, the attention of those present at the 'Mecca of Cricket'.

Figure 1.1: An article from Sportstar.

knowledge but test if machines can perform *applied* language tasks – for example, being able to ask and answer questions about a range of topics or being able to summarize articles. While such tasks could and often do require the kind of linguistic and factual knowledge we talked about, they could also require other cognitive skills closely aligned to core NLP. For example, news summarization might require the reader to pick out and articulate the most important parts of an article often in ways that are interesting to readers. In other words, apart from linguistic and factual background knowledge, performing these applied tasks could require task-specific knowledge.

1.1.1 Self-supervised Models: Pre-training and Fine-tuning

Self-supervised models use the natural supervision provided by unlabeled text for better initialization of parameters. Much of recent progress on both core and applied fronts has been enabled with the creation of large self-supervised models and downstream task datasets. The typical training pipeline works as follows:

self-supervised models are first "pre-trained" i.e. trained on unlabeled – and often downstream task-agnostic – text and then "fine-tuned" i.e. trained further on downstream task data.

Most pre-training objectives are denoising autoencoders that are trained to reconstruct text where a random subset of the words has been noised or hidden. For example, training left-to-right language models involve predicting a token from a given prefix and each time step. Likewise, masked language modeling involves predicting a small subset of tokens (replaced with special masked tokens) from the rest of the context. Recent work has shown that pre-trained models are able to encode a considerable amount of factual and linguistic knowledge ranging from word and entity relations to syntax Rogers et al. [2020].

What helps self-supervised models capture at least a part of world knowledge that is need for robust language processing? State of the art models contain millions of parameters and were pre-trained over large corpora using several thousand GPUs. For example, RoBERTa Liu et al. [2019c], with 350 million parameters was trained on 160 GB of text using 1024 GPUs. The scale of pre-trained models has continued to increase along with a pronounced increase performance over a wide variety of tasks.

In spite of this large scale pre-training, the best performing models have to be further fine-tuned on downstream tasks to achieve state of the art performance. For example, SQuAD Rajpurkar et al. [2016] one of the most popular NLP datasets for reading comprehension contains around 100,000 questions. Large scale training is effective but expensive. Pre-training itself is compute intensive while creating large high-quality datasets for downstream tasks requires considerable expertise and quality control. Both steps in turn translate into monetary cost.

The aim of this thesis is to provide training paradigms for efficient self-supervised models, and to minimize the cost of training these models. Specifically, we have the following goals:

1. Designing efficient pre-training methods to capture linguistic and world knowledge

2. Enabling better downstream performance with fewer human-labeled examples

1.2 Background

We categorize related work into three main groups: (*i*) general purpose language representations which encode world knowledge, (*ii*) integrating additional background knowledge for specific downstream tasks,

and (*iii*) data augmentation.

1.2.1 General Purpose Language Representations

Training general purpose language representations has a long history stretching as far back as matrix factorization methods like latent semantic analysis (LSA) Deerwester et al. [1990]. Subsequent work used single-word embeddings such as Glove Pennington et al. [2014] and Word2vec Mikolov et al. [2013a], trained following the Distributional Hypothesis Harris [1954]. Pre-trained contextualized word representations that can be trained from unlabeled text Dai and Le [2015]; Melamud et al. [2016]; Peters et al. [2018] have had immense impact on NLP lately, particularly as methods for initializing a large model before fine-tuning it for a specific task Howard and Ruder [2018]; Radford et al. [2018]; Devlin et al. [2018]. Beyond differences in model hyperparameters and corpora, these methods mainly differ in their pre-training tasks and loss functions, with a considerable amount of contemporary literature proposing augmentations of BERT's masked language modeling (MLM) objective. We build upon this work in two ways: (1) in Chapter 3, we extend MLM-based methods for pre-training span representations consistently outperforming BERT, with the largest gains on span selection tasks such as question answering and coreference resolution, and (2) in Chapter 3, we re-use BERT-style representations for jointly encoding the input with dynamically-retrieved textual background knowledge.

1.2.2 Task-specific Background Knowledge

Earlier work Ratinov and Roth [2009]; Nakashole and Mitchell [2015] combined features over the given task data with hand-engineered features over knowledge repositories. Other forms of external knowledge include relational knowledge between word or entity pairs, typically integrated via embeddings from structured knowledge graphs (KGs) Yang and Mitchell [2017]; Bauer et al. [2018]; Mihaylov and Frank [2018]; Wang and Jiang [2019] or via word pair embeddings trained from text Joshi et al. [2019a]. Weissenborn et al. [2017] used a specialized architecture to integrate background knowledge from ConceptNet and Wikipedia entity descriptions. For open-domain QA, recent works Sun et al. [2019a]; Xiong et al. [2019] jointly reasoned over text and KGs, via specialized graph-based architectures for defining the flow of information between them.

Most relevant to ours is work building upon these powerful pretrained representations, and further integrating external knowledge. Recent work focuses on refining pretrained contextualized representations using entity or triple embeddings from structured KGs Peters et al. [2019]; Yang et al. [2019a]; Zhang et al. [2019b]. The KG embeddings are trained separately (often to predict links in the KG), and knowledge from KG is fused with deep Transformer representations via special-purpose architectures. Some of these prior works also pre-train the knowledge fusion layers from unlabeled text through self-supervised objectives Zhang et al. [2019b]; Peters et al. [2019]. Instead of separately encoding structured KBs, and then attending to their single-vector embeddings, we explore directly using wider-coverage textual encyclopedic background knowledge (Chapter 4). This enables direct application of a pretrained deep Transformer (RoBERTa) for jointly contextualizing input text and background knowledge. We showed background knowledge integration can be further improved by additional knowledge-augmented self-supervised pretraining.

In concurrent work, Liu et al. [2019a] augment text with relevant triples from a structured KB. They process triples as word sequences using BERT with a special-purpose attention masking strategy. This allows the model to partially re-use BERT for encoding and integrating the structured knowledge. Our work uses wider-coverage textual sources instead and shows the power of additional knowledge-tailored self-supervised pretraining.

1.2.3 Data Augmentation

Data augmentation is a popular technique with a large body of work Wang and Yang [2015]; Jia and Liang [2016] among others. Recent work has explored model generated data augmentation for a range of tasks including text classification Anaby-Tavor et al. [2020]; Schick and Schütze [2021a], question answering Alberti et al. [2019], common-sense reasoning Yang et al. [2020], and machine translation Sennrich et al. [2016]. A common problem with model-generated data augmentation is the quality of the synthetic data. Attempts to remedy this have focused on variations of consistency Xie et al. [2019] for a given task–such as round-trip consistency of question generation and answer prediction Alberti et al. [2019]; Puri et al. [2020] for QA or between source and targets in summarization Fabbri et al. [2021]. Lee et al. [2021] focus on generating synthetic data for underrpresented or few-shot slices. Task augmentation Vu et al. [2021] generates

data in the target domain by using a model trained on the auxiliary task of natural language inference. In contrast, in Chapter 5, we focus on a general framework for mining *naturally occurring* data for multiple tasks using supervision from a small labeled seed set.

1.3 Outline

This thesis presents solutions which can be divided into three broad paradigms – (a) self-supervised objectives, (b) model scale, and (c) new types of data. As we will show in the following chapters, *self-supervised objectives* could have a large influence on the form of knowledge that is acquired during pre-training. Moreover, efficient objectives directly enable *model scale* both in terms of data and parameters. Finally, the training *data* and the kind of supervision derived from it itself dictates how well a model can learn different kinds of downstream tasks, and could reduce the amount of human-labeling required for human performance.

1.3.1 Efficient pre-training methods to capture linguistic and world knowledge

One justification, particularly for large models, is that the kind of linguistic and factual knowledge we seek to encode is spread over millions of documents making large scale training arguably inevitable. While alternate paradigms (e.g. non-parametric methods) exist, large scale pre-training has driven much of empirical progress over the last few years. We argue that training paradigms for self-supervised models should aim to encode diverse kinds of knowledge across corpora of of varying domains. We seek to answer the following questions:

1. How can be design pre-training objectives to capture the kinds of knowledge we care about?
2. How can we scale up models in terms of data and parameters to maximize the coverage of training corpora?

Chapter 2 focuses on self-supervised objectives for relational knowledge i.e. reasoning about relationships between pairs of words. Such knowledge is crucial for several NLP tasks. In NLI, for example, given the premise "*golf is prohibitively expensive*", inferring that the hypothesis "*golf is a cheap pastime*" is a contradiction requires one to know that *expensive* and *cheap* are antonyms. Our word-pair representations are learned by modeling the three-way co-occurrence between words (x, y) and the context c that ties them

together. For example, it is easy to infer that the words "cheap" and "expensive" are antonyms based on co-occuring contexts such as "either x or y" We show that they can be learned with word *pair* vectors (`pair2vec`), which significantly improve performance when added to existing cross-sentence attention mechanisms.

Chapter 3 focuses scalable efficient objectives which encode a wider variety of linguistic and world knowledge into model parameters. We present SpanBERT, a pre-training method that is designed to better represent and predict spans of text. Unlike token-based MLM, span based objectives focus on predicting longer spans. The intuition is target more efficient pre-training by preventing the model from making predictions based on immediate short-term context. We show that span-based pre-training improves the state of the art for a range of NLP tasks including those which do not require explicit span based reasoning.

Chapter 4 focuses integrating dynamically retrieved textual knowledge. Specifically, even large scale representations are not able to preserve all factual knowledge they have "read" during pre-training due to the long tail of entity and event-specific information. Training models to integrate background knowledge during pre-training is especially useful for downstream tasks which require reasoning over this long tail. Taken together, Chapters 3 and 4 focus on maximizing the coverage of knowledge in form and content.

1.3.2 Enabling better downstream performance with fewer human-labeled examples

As we discussed in Section 1.1, large scale pre-training, the best performing models have to be further fine-tuned on downstream tasks to achieve state of the art performance.

Chapter 5 focuses on reducing the amount of downstream data by mining input-output pairs (and thus obtaining direct task-level supervision) from corpora using supervision from very few labeled examples. We mine naturally-occurring examples from a large corpus using supervision from a small seed set of only 100 examples. Our method provides a way to collect more data using very few examples. Yet, unlike model generated data augmentation, we mine high-quality human-authored data which is less susceptible to the limitations synthetic data.

Chapter 2

pair2vec: Compositional Word-Pair Embeddings for Cross-Sentence Inference

This chapter discusses work originally published in Joshi et al. [2019a]

Reasoning about relationships between pairs of words is crucial for cross sentence inference problems such as question answering (QA) and natural language inference (NLI). In NLI, for example, given the premise "*golf is prohibitively expensive*", inferring that the hypothesis "*golf is a cheap pastime*" is a contradiction requires one to know that *expensive* and *cheap* are antonyms. Recent work Glockner et al. [2018] has shown that current models, which rely heavily on unsupervised single-word embeddings, struggle to learn such relationships. In this chapter, we show that they can be learned with word *pair* vectors (`pair2vec`[1]), which are trained unsupervised, and which significantly improve performance when added to existing cross-sentence attention mechanisms.

Unlike single-word representations, which typically model the co-occurrence of a target word x with its context c, our word-pair representations are learned by modeling the three-way co-occurrence between words (x, y) and the context c that ties them together, as seen in Table 2.1. While similar training signals have been used to learn models for ontology construction Hearst [1992]; Snow et al. [2005]; Turney [2005]; Shwartz et al. [2016] and knowledge base completion Riedel et al. [2013], this chapter shows, for the first time, that large scale learning of pairwise embeddings can be used to directly improve the performance of

[1] `https://github.com/mandarjoshi90/pair2vec`

X	Y	Contexts
hot	*cold*	with **X** and **Y** baths too **X** or too **Y** neither **X** nor **Y**
Portland	*Oregon*	in **X**, **Y** the **X** metropolitan area in **Y** **X** International Airport in **Y**
crop	*wheat*	food **X** are maize, **Y**, etc dry **X**, such as **Y**, more **X** circles appeared in **Y** fields
Android	*Google*	**X** OS comes with **Y** play the **X** team at **Y** **X** is developed by **Y**

Table 2.1: Example word pairs with contexts.

neural cross-sentence inference models.

More specifically, we train a feedforward network $R(x, y)$ that learns representations for the individual words x and y, as well as how to compose them into a single vector. Training is done by maximizing a generalized notion of the pointwise mutual information (PMI) among x, y, and their context c using a variant of negative sampling Mikolov et al. [2013a]. Making $R(x, y)$ a compositional function on individual words alleviates the sparsity that necessarily comes with embedding pairs of words, even at a very large scale.

We show that our embeddings can be added to existing cross-sentence inference models, such as BiDAF++ Seo et al. [2017]; Clark and Gardner [2018] for QA and ESIM Chen et al. [2017] for NLI. Instead of changing the word embeddings that are fed into the encoder, we add the pretrained *pair* representations to *higher layers* in the network where cross sentence attention mechanisms are used. This allows the model to use the background knowledge that the pair embeddings implicitly encode to reason about the likely relationships between the pairs of words it aligns.

Experiments show that simply adding our word-pair embeddings to existing high-performing models, which already use ELMo Peters et al. [2018], results in sizable gains. We show 2.72 F1 points over the BiDAF++ model Clark and Gardner [2018] on SQuAD 2.0 Rajpurkar et al. [2018], as well as a 1.3 point gain over ESIM Chen et al. [2017] on MultiNLI Williams et al. [2018a]. Additionally, our approach generalizes well to adversarial examples, with a 6-7% F1 increase on adversarial SQuAD Jia and Liang [2017] and a 8.8% gain on the Glockner et al. [2018] NLI benchmark. An analysis of `pair2vec` on word analogies

suggests that it complements the information in single-word representations, especially for encyclopedic and lexicographic relations.

2.1 Self-supervised Pretraining

Extending the distributional hypothesis to word pairs, we posit that similar word *pairs* tend to occur in similar contexts, and that the contexts provide strong clues about the likely relationships that hold between the words (see Table 2.1). We assume a dataset of (x, y, c) triplets, where each instance depicts a word pair (x, y) and the context c in which they appeared. We learn two compositional representation functions, $R(x, y)$ and $C(c)$, to encode the pair and the context, respectively, as d-dimensional vectors (Section 2.1.1). The functions are trained using a variant of negative sampling, which tries to embed word pairs (x, y) close to the contexts c with which they appeared (Section 2.1.2).

2.1.1 Representation

Our word-pair and context representations are both fixed-length vectors, composed from individual words. The word-pair representation function $R(x, y)$ first embeds and normalizes the individual words with a shared lookup matrix E_a:

$$\mathbf{x} = \frac{E_a(x)}{\|E_a(x)\|} \qquad \mathbf{y} = \frac{E_a(y)}{\|E_a(y)\|}$$

These vectors, along with their element-wise product, are fed into a four-layer perceptron:

$$R(x, y) = MLP^4(\mathbf{x}, \mathbf{y}, \mathbf{x} \circ \mathbf{y})$$

The context $c = c_1 ... c_n$ is encoded as a d-dimensional vector using the function $C(c)$. $C(c)$ embeds each token c_i with a lookup matrix E_c, contextualizes it with a single-layer Bi-LSTM, and then aggregates the

entire context with attentive pooling:

$$\mathbf{c}_i = E_c(c_i)$$

$$\mathbf{h_1}...\mathbf{h_n} = \text{BiLSTM}(\mathbf{c}_1...\mathbf{c}_n)$$

$$w = \text{softmax}_i(\mathbf{kh_i})$$

$$C(c) = \sum_i w_i \mathbf{W}\mathbf{h}_i$$

where $\mathbf{W} \in \mathbb{R}^{d \times d}$ and $\mathbf{k} \in \mathbb{R}^d$. All parameters, including the lookup tables E_a and E_c, are trained.

Our representation is similar to two recently-proposed frameworks by Washio and Kato [2018a,b], but differs in that: (1) they use dependency paths as context, while we use surface form; (2) they encode the context as either a lookup table or the last state of a unidirectional LSTM. We also use a different objective, which we discuss next.

2.1.2 Objective

To optimize our representation functions, we consider two variants of negative sampling Mikolov et al. [2013a]: bivariate and multivariate. The original bivariate objective models the two-way distribution of context and (monolithic) word pair co-occurrences, while our multivariate extension models the three-way distribution of word-word-context co-occurrences. We further augment the multivariate objective with typed sampling to upsample harder negative examples. We discuss the impact of the bivariate and multivariate objectives (and other components) in Section 2.3.3.

Bivariate	$J_{2NS}(x,y,c) = \log\sigma\left(R(x,y)\cdot C(c)\right) + \sum_{i=1}^{k_c}\log\sigma\left(-R(x,y)\cdot C(c_i^N)\right)$
Multivariate	$J_{3NS}(x,y,c) = J_{2NS}(x,y,c) + \sum_{i=1}^{k_x}\log\sigma\left(-R(x_i^N,y)\cdot C(c)\right) + \sum_{i=1}^{k_y}\log\sigma\left(-R(x,y_i^N)\cdot C(c)\right)$

Table 2.2: The bivariate and multivariate negative sampling objectives. The superscript ^{N} marks randomly sampled components, with k_* being the negative sample size per instance. The equations present per-instance objectives.

Bivariate Negative Sampling Our objective aspires to make $R(x,y)$ and $C(c)$ similar (have high inner products) for (x,y,c) that were observed together in the data. At the same time, we wish to keep our pair

vectors *dis*-similar from random context vectors. In a straightforward application of the original (bivariate) negative sampling objective, we could generate a negative example from each observed (x, y, c) instance by replacing the original context c with a randomly-sampled context c^N (Table 2.2, J_{2NS}).

Assuming that the negative contexts are sampled from the empirical distribution $P(\cdot, \cdot, c)$ (with $P(x, y, c)$ being the portion of (x, y, c) instances in the dataset), we can follow Levy and Goldberg [2014] to show that this objective converges into the pointwise mutual information (PMI) between the word pair and the context.

$$R(x, y) \cdot C(c) = \log \frac{P(x, y, c)}{k_c P(x, y, \cdot) P(\cdot, \cdot, c)}$$

This objective mainly captures co-occurrences of monolithic pairs and contexts, and is limited by the fact that the training data, by construction, only contains pairs occurring within a sentence. For better generalization to cross-sentence tasks, where the pair distribution differs from that of the training data, we need a multivariate objective that captures the full three-way (x, y, c) interaction.

Multivariate Negative Sampling We introduce negative sampling of target words, x and y, in addition to negative sampling of contexts c (Table 2.2, J_{3NS}). Our new objective also converges to a novel multivariate generalization of PMI, different from previous PMI extensions that were inspired by information theory Van de Cruys [2011] and heuristics Jameel et al. [2018].[2] Following Levy and Goldberg (2014), we can show that when replacing target words in addition to contexts, our objective will converge[3] to the optimal value in Equation 2.1:

$$R(x, y) \cdot C(c) = \log \frac{P(x, y, c)}{Z_{x,y,c}} \tag{2.1}$$

where $Z_{x,y,c}$, the denominator, is:

$$Z_{x,y,c} = k_c P(\cdot, \cdot, c) P(x, y, \cdot) + k_x P(x, \cdot, \cdot) P(\cdot, y, c) + k_y P(\cdot, y, \cdot) P(x, \cdot, c) \tag{2.2}$$

This optimal value deviates from previous generalizations of PMI by having a linear mixture of marginal

[2] See supplementary material for their exact formulations.

[3] A full proof is provided in the supplementary material.

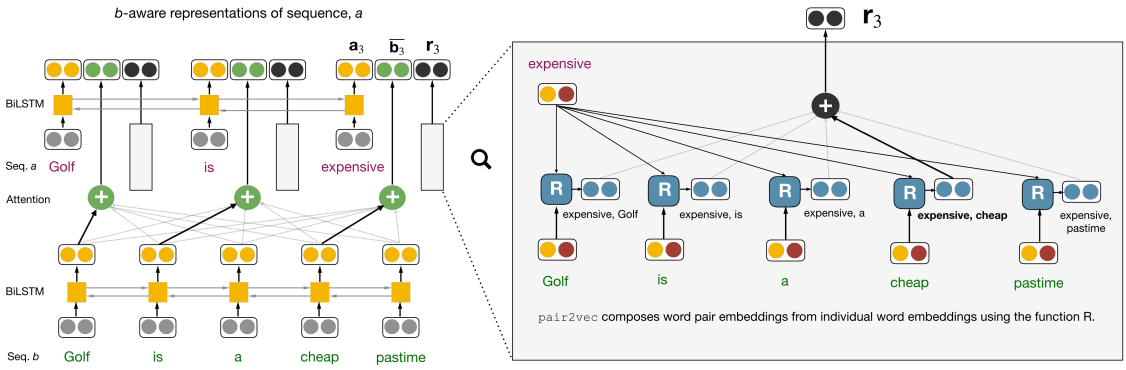

Figure 2.1: A typical architecture of a cross-sentence inference model (left), and how `pair2vec` is added to it (right). Given two sequences, a and b, existing models create b-aware representations of words in a. For any word a_i, this typically involves the BiLSTM representation of word a_i ($\mathbf{a}_i$), and an attention-weighted sum over b's BiLSTM states with a_i as the query ($\overline{\mathbf{b}}_i$). To these, we add the word-pair representation of a_i and each word in b, weighted by attention ($\mathbf{r}_i$). Thicker attention arrows indicate stronger word pair alignments (e.g. *cheap*, *expensive*).

probability products in its denominator. By introducing terms such as $P(x, \cdot, c)$ and $P(\cdot, y, c)$, the objective penalizes spurious correlations between words and contexts that disregard the other target word. For example, it would assign the pattern "*X is a Y*" a high score with (*banana*, *fruit*), but a lower score with (*cat*, *fruit*).

Typed Sampling In multivariate negative sampling, we typically replace x and y by sampling from their unigram distributions. In addition to this, we also sample uniformly from the top 100 words according to cosine similarity using distributional word vectors. This is done to encourage the model to learn relations between specific instances as opposed to more general types. For example, using *California* as a negative sample for *Oregon* helps the model to learn that the pattern "*X is located in Y*" fits the pair (*Portland*, *Oregon*), but not the pair (*Portland*, *California*). Similar adversarial constraints were used in knowledge base completion Toutanova et al. [2015] and word embeddings Li et al. [2017].[4]

2.2 Integrating `pair2vec` into Models

We first present a general outline for incorporating `pair2vec` into attention-based architectures, and then discuss changes made to BiDAF++ and ESIM. The key idea is to inject our pairwise representations into the attention layer by reusing the cross-sentence attention weights. In addition to attentive pooling over single word representations, we also pool over cross-sentence word pair embeddings (Figure 2.1).

2.2.1 General Approach

Pair Representation We assume that we are given two sequences $a = a_1...a_n$ and $b = b_1...b_m$. We represent the word-pair embeddings between a and b using the pretrained `pair2vec` model as:

$$\mathbf{r}_{i,j} = \left[\frac{R(a_i, b_j)}{\|R(a_i, b_j)\|}; \frac{R(b_j, a_i)}{\|R(b_j, a_i)\|} \right] \tag{2.3}$$

We include embeddings in both directions, $R(a_i, b_j)$ and $R(b_j, a_i)$, because the many relations can be expressed in both directions; e.g., hypernymy can be expressed via "*X is a type of Y*" as well as "*Y such as X*". We take the L_2 normalization of each direction's pair embedding because the heavy-tailed distribution of word pairs results in significant variance of their norms.

Base Model Let $\mathbf{a}_1...\mathbf{a}_n$ and $\mathbf{b}_1...\mathbf{b}_m$ be the vector representations of sequences a and b, as produced by the input encoder (e.g. ELMo embeddings contextualized with model-specific BiLSTMs). Furthermore, we assume that the base model computes soft word alignments between a and b via co-attention (2.4, 2.5), which are then used to compute b-aware representations of a:

$$s_{i,j} = f_{att}(\mathbf{a}_i, \mathbf{b}_j) \tag{2.4}$$

$$\alpha = \text{softmax}_j(s_{i,j}) \tag{2.5}$$

$$\overline{\mathbf{b}}_i = \sum_{j=0}^{m} \alpha_{i,j} \mathbf{b}_j \tag{2.6}$$

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i \right] \tag{2.7}$$

[4] Applying typed sampling also changes the value to which our objective will converge, and will replace the unigram probabilities in Equation (2.2) to reflect the type-based distribution.

The symmetric term $\mathbf{b}_j^{inf}$ is defined analogously. We refer to $\mathbf{a}^{inf}$ and $\mathbf{b}^{inf}$ as the inputs to the *inference* layer, since this layer computes some function over aligned word pairs, typically via a feedforward network and LSTMs. The inference layer is followed by aggregation and output layers.

Injecting `pair2vec` We conjecture that the inference layer effectively learns word-pair relationships from training data, and it should, therefore, help to augment its input with `pair2vec`. We augment $\mathbf{a}_i^{inf}$ (2.7) with the pair vectors $\mathbf{r}_{i,j}$ (2.3) by concatenating a weighted average of the pair vectors $r_{i,j}$ involving a_i, where the weights are the same $\alpha_{i,j}$ computed via attention in (2.5):

$$\mathbf{r}_i = \sum_j \alpha_{i,j} \mathbf{r}_{i,j} \tag{2.8}$$

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i; \mathbf{r}_i\right] \tag{2.9}$$

The symmetric term $\mathbf{b}_j^{inf}$ is defined analogously.

2.2.2 Question Answering

We augment the inference layer in the BiDAF++ model with `pair2vec`. BiDAF++ is an improved version of the BiDAFNoAnswer Seo et al. [2017]; Levy et al. [2017] which includes self-attention and ELMo embeddings from Peters et al. [2018]. We found this variant to be stronger than the baselines presented in Rajpurkar et al. [2018] by over 2.5 F1. We use BiDAF++ as a baseline since its architecture is typical for QA systems, and, until recently, was state-of-the-art on SQuAD 2.0 and other benchmarks.

BiDAF++ Let $\mathbf{a}$ and $\mathbf{b}$ be the outputs of the passage and question encoders respectively (in place of the standard $\mathbf{p}$ and $\mathbf{q}$ notations). The inference layer's inputs $\mathbf{a}_i^{inf}$ are defined similarly to the generic model's in (2.7), but also contain an aggregation of the elements in $\mathbf{a}$, with better-aligned elements receiving larger

weights:

$$\mu = \text{softmax}_i(\max_j s_{i,j}) \tag{2.10}$$

$$\hat{\mathbf{a}}_i = \sum_i \mu_i \mathbf{a}_i \tag{2.11}$$

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i; \mathbf{a}_i \circ \overline{\mathbf{b}}_i; \hat{\mathbf{a}}\right] \tag{2.12}$$

In the later layers, $\mathbf{a}^{inf}$ is recontextualized using a BiGRU and self attention. Finally a prediction layer predicts the start and end tokens.

BiDAF++ with `pair2vec` To add our pair vectors, we simply concatenate $\mathbf{r}_i$ (2.3) to $\mathbf{a}_i^{inf}$ (2.12):

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i; \mathbf{a}_i \circ \overline{\mathbf{b}}_i; \hat{\mathbf{a}}; \mathbf{r}_i\right] \tag{2.13}$$

2.2.3 Natural Language Inference

For NLI, we augment the ESIM model Chen et al. [2017], which was previously state-of-the-art on both SNLI Bowman et al. [2015] and MultiNLI Williams et al. [2018a] benchmarks.

ESIM Let $\mathbf{a}$ and $\mathbf{b}$ be the outputs of the premise and hypothesis encoders respectively (in place of the standard $\mathbf{p}$ and $\mathbf{h}$ notations). The inference layer's inputs $\mathbf{a}_i^{inf}$ (and $\mathbf{b}_j^{inf}$) are defined similarly to the generic model's in (2.7):

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i; \mathbf{a}_i \circ \overline{\mathbf{b}}_i; \mathbf{a}_i - \overline{\mathbf{b}}_i\right] \tag{2.14}$$

In the later layers, $\mathbf{a}^{inf}$ and $\mathbf{b}^{inf}$ are projected, recontextualized, and converted to a fixed-length vector for each sentence using multiple pooling schemes. These vectors are then passed on to an output layer, which predicts the class.

ESIM with `pair2vec` To add our pair vectors, we simply concatenate $\mathbf{r}_i$ (2.3) to $\mathbf{a}_i^{inf}$ (2.14):

$$\mathbf{a}_i^{inf} = \left[\mathbf{a}_i; \overline{\mathbf{b}}_i; \mathbf{a}_i \circ \overline{\mathbf{b}}_i; \mathbf{a}_i - \overline{\mathbf{b}}_i; \mathbf{r}_i\right] \tag{2.15}$$

A similar augmentation of ESIM was recently proposed in KIM Chen et al. [2018]. However, their pair vectors are composed of WordNet features, while our pair embeddings are learned directly from text.

2.3 Experiments

For experiments on QA (Section 2.3.1) and NLI (Section 2.3.2), we use our full model which includes multivariate and typed negative sampling. We discuss ablations in Section 2.3.3

Data We use the January 2018 dump of English Wikipedia, containing 96M sentences to train `pair2vec`. We restrict the vocabulary to the 100K most frequent words. Preprocessing removes all out-of-vocabulary words in the corpus. We consider each word pair within a window of 5 in the preprocessed corpus, and subsample[5] instances based on pair probability with a threshold of $5 \cdot 10^{-7}$. We define the context as one word each to the left and right, and all the words in between each pair, replacing both target words with placeholders X and Y (see Table 2.1). More details can be found in the supplementary material.

2.3.1 Question Answering

Benchmark		BiDAF	+ `pair2vec`	Δ
SQuAD 2.0	EM	65.66	68.02	+2.36
	F1	68.86	71.58	+2.72
AddSent	EM	37.50	44.20	+6.70
	F1	42.55	49.69	+7.14
AddOneSent	EM	48.20	53.30	+5.10
	F1	54.02	60.13	+6.11

Table 2.3: Performance on SQuAD 2.0 and adversarial SQuAD (AddSent and AddOneSent) benchmarks, with and without `pair2vec`. All models have ELMo.

[5]Like in `word2vec`, subsampling reduces the size of the dataset and speeds up training. For this, we define the word pair probability as the product of unigram probabilities.

Benchmark	ESIM	+ pair2vec	Δ
Matched	79.68	81.03	+1.35
Mismatched	78.80	80.12	+1.32

Table 2.4: Performance on MultiNLI, with and without pair2vec. All models have ELMo.

Model	Accuracy
Rule-based Models	
WordNet Baseline	85.8
Models with GloVe	
ESIM Chen et al. [2017]	77.0
KIM Chen et al. [2018]	87.7
ESIM + pair2vec	**92.9**
Models with ELMo	
ESIM Peters et al. [2018]	84.6
ESIM + pair2vec	**93.4**

Table 2.5: Performance on the adversarial NLI test set of Glockner et al. [2018].

We experiment on the SQuAD 2.0 QA benchmark Rajpurkar et al. [2018], as well as the adversarial datasets of SQuAD 1.1 Rajpurkar et al. [2016]; Jia and Liang [2017]. Table 2.3 shows the performance of BiDAF++, with ELMo , before and after adding pair2vec. Experiments on SQuAD 2.0 show that our pair representations improve performance by 2.72 F1. Moreover, adding pair2vec also results in better generalization on the adversarial SQuAD datasets with gains of 7.14 and 6.11 F1.

2.3.2 Natural Language Inference

We report the performance of our model on MultiNLI and the adversarial test set from Glockner et al. [2018] in Table 2.5. We outperform the ESIM + ELMo baseline by 1.3% on the matched and mismatched portions of the dataset.

We also record a gain of 8.8% absolute over ESIM on the Glockner et al. [2018] dataset, setting a new state of the art. Following standard practice Glockner et al. [2018], we train all models on a combination of SNLI Bowman et al. [2015] and MultiNLI. Glockner et al. [2018] show that with the exception of KIM Chen et al. [2018], which uses WordNet features, several NLI models fail to generalize to this setting which involves lexical inference. For a fair comparison with KIM on the Glockner test set, we replace ELMo with GLoVE embeddings, and still outperform KIM by almost halving the error rate.

Model	EM (Δ)	F1 (Δ)
`pair2vec` (Full Model)	69.20	72.68
Composition: 2 Layers	68.35 (-0.85)	71.65 (-1.03)
Composition: Multiply	67.10 (-2.20)	70.20 (-2.48)
Objective: Bivariate NS	68.63 (-0.57)	71.98 (-0.70)
Unsupervised: Pair Dist	67.07 (-2.13)	70.24 (-2.44)
No `pair2vec` (BiDAF)	66.66 (-2.54)	69.90 (-2.78)

Table 2.6: Ablations on the SQuAD 2.0 development set show that argument sampling as well as using a deeper composition function are useful.

2.3.3 Ablations

Ablating parts of `pair2vec` shows that all components of the model (Section 2.1) are useful. We ablate each component and report the EM and F1 on the development set of SQuAD 2.0 (Table 2.6). The full model, which uses a 4-layer MLP for $R(x, y)$ and trains with multivariate negative sampling, achieves the highest F1 of 72.68.

We experiment with two alternative composition functions, a 2-layer MLP (*Composition: 2 Layers*) and element-wise multiplication (*Composition: Multiply*), which yield significantly smaller gains over the baseline BiDAF++ model. This demonstrates the need for a deep composition function. Eliminating sampling of target words (x, y) from the objective (*Objective: Bivariate NS*) results in a drop of 0.7 F1, accounting for about a quarter of the overall gain. This suggests that while the bulk of the signal is mined from the pair-context interactions, there is also valuable information in other interactions as well.

We also test whether specific pre-training of word *pair* representations is useful by replacing `pair2vec` embeddings with the vector offsets of pre-trained word embeddings (*Unsupervised: Pair Dist*). We follow the PairDistance method for word analogies Mikolov et al. [2013b], and represent the pair (x, y) as the L2 normalized difference of single-word vectors: $(\mathbf{x} - \mathbf{y})/\|\mathbf{x} - \mathbf{y}\|$. We use the same fastText Bojanowski et al. [2017] word vectors with which we initialized `pair2vec` before training. We observe a gain of only 0.34 F1 over the baseline.

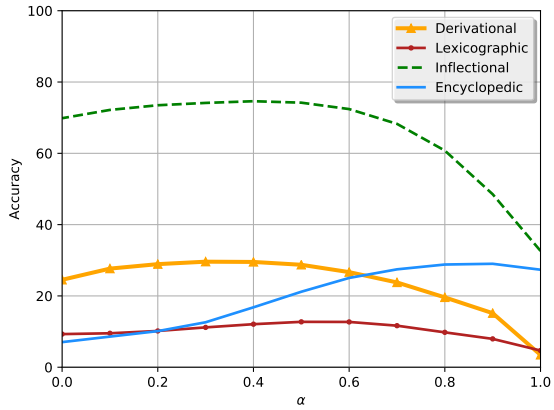

Figure 2.2: Accuracy as a function of the interpolation parameter α (see Eq. (2.16)). The α=0 configuration relies only on fastText Bojanowski et al. [2017], while α=1 reflects `pair2vec`.

Relation	`3CosAdd`	`+pair2vec`	α^*
Country:Capital	1.2	86.1	0.9
Name:Occupation	1.8	44.6	0.8
Name:Nationality	0.1	42.0	0.9
UK City:County	0.7	31.7	1.0
Country:Language	4.0	28.4	0.8
Verb 3pSg:Ved	49.1	61.7	0.6
Verb Ving:Ved	61.1	73.3	0.5
Verb Inf:Ved	58.5	70.1	0.5
Noun+less	4.8	16.0	0.2
Substance Meronym	3.8	14.5	0.6

Table 2.7: The top 10 analogy relations for which interpolating with `pair2vec` improves performance. α^* is the optimal interpolation parameter for each relation.

2.4 Analysis

In Section 2.3, we showed that `pair2vec` adds information complementary to single-word representations like ELMo. Here, we ask what this extra information is, and try to characterize which word relations are better captured by `pair2vec`. To that end, we evaluate performance on a word analogy dataset with over 40 different relation types (Section 2.4.1), and observe how `pair2vec` fills hand-crafted relation patterns (Section 2.4.2).

2.4.1 Quantitative Analysis: Word Analogies

Word Analogy Dataset Given a word pair (a, b) and word x, the word analogy task involves predicting a word y such that $a : b :: x : y$. We use the Bigger Analogy Test Set [BATS, Gladkova et al., 2016]

Relation	Context	X	Y (Top 3)
Antonymy/Exclusion	either X or Y	accept hard	*reject*, *refuse*, recognise *soft*, *brittle*, *polished*
Hypernymy	including X and other Y	copper google	ones, *metals*, *mines* apps, *browsers*, *searches*
Hyponymy	X like Y	cities browsers	solaris, *speyer*, *medina* *chrome*, *firefox*, *netscape*
Co-hyponymy	, X , Y ,	copper google	*malachite*, *flint*, ivory *microsoft*, *bing*, *yahoo*
City-State	in X , Y .	portland dallas	*oregon*, *maine*, *dorset* *tx*, *texas*, va
City-City	from X to Y .	portland dallas	*salem*, *astoria*, ogdensburg *denton*, *allatoona*, *addison*
Profession	X , a famous Y ,	ronaldo monet	*footballer*, portuguese, *player* *painter*, painting, butterfly

Table 2.8: Given a context c and a word x, we select the top 3 words y from the entire vocabulary using our scoring function $R(x, y) \cdot C(c)$. The analysis suggests that the model tends to rank correct matches (italics) over others.

which contains four groups of relations: encyclopedic semantics (e.g., person-profession as in *Einstein-physicist*), lexicographic semantics (e.g., antonymy as in *cheap-expensive*), derivational morphology (e.g., noun forms as in *oblige-obligation*), and inflectional morphology (e.g., noun-plural as in *bird-birds*). Each group contains 10 sub-relations.

Method We interpolate `pair2vec` and `3CosAdd` Mikolov et al. [2013b]; Levy et al. [2014] scores on fastText embeddings, as follows:

$$\begin{aligned} \text{score}(y) = \alpha \cdot \cos(\mathbf{r}_{a,b}, \mathbf{r}_{x,y}) \\ + (1 - \alpha) \cdot \cos(\mathbf{b} - \mathbf{a} + \mathbf{x}, \mathbf{y}) \end{aligned} \tag{2.16}$$

where $\mathbf{a}$, $\mathbf{b}$, $\mathbf{x}$, and $\mathbf{y}$ represent fastText embeddings[6] and $\mathbf{r}_{a,b}$, $\mathbf{r}_{x,y}$ represent the `pair2vec` embedding for the word pairs (a, b) and (x, y), respectively; α is the linear interpolation parameter. Following prior work Mikolov et al. [2013b], we return the highest-scoring y in the entire vocabulary, excluding the given words a, b, and x.

[6]The fastText embeddings in the analysis were retrained using the same Wikipedia corpus used to train `pair2vec` to control for the corpus when comparing the two methods.

Results Figure 2.2 shows how the accuracy on each category of relations varies with α. For all four groups, adding `pair2vec` to `3CosAdd` results in significant gains. In particular, the biggest relative improvements are observed for encyclopedic (356%) and lexicographic (51%) relations.

Table 2.7 shows the specific relations in which `pair2vec` made the largest absolute impact. The gains are particularly significant for relations where fastText embeddings provide limited signal. For example, the accuracy for *substance meronyms* goes from 3.8% to 14.5%. In some cases, there is also a synergistic effect; for instance, in *noun+less*, `pair2vec` alone scored 0% accuracy, but mixing it with `3CosAdd`, which got 4.8% on its own, yielded 16% accuracy.

These results, alongside our experiments in Section 2.3, strongly suggest that `pair2vec` encodes information complementary to that in single-word embedding methods such as fastText and ELMo.

2.4.2 Qualitative Analysis: Slot Filling

To further explore how `pair2vec` encodes such complementary information, we consider a setting similar to that of knowledge base completion: given a Hearst-like context pattern c and a single word x, predict the other word y from the entire vocabulary. We rank candidate words y based on the scoring function in our training objective: $R(x, y) \cdot C(c)$. We use a fixed set of example relations and manually define their predictive context patterns and a small set of candidate words x.

Table 2.8 shows the top three y words. The model embeds (x, y) pairs close to contexts that reflect their relationship. For example, substituting *Portland* in the city-state pattern ("*in X, Y.*"), the top two words are *Oregon* and *Maine*, both US states with cities named Portland. When used with the city-city pattern ("*from X to Y.*"), the top two words are *Salem* and *Astoria*, both cities in Oregon. The word-context interaction often captures multiple relations; for example, *Monet* is used to refer to the painter (*profession*) as well as his paintings.

As intended, `pair2vec` captures the three-way word-word-context interaction, and not just the two-way word-context interaction (as in single-word embeddings). This profound difference allows `pair2vec` to complement single-word embeddings with additional information.

2.5 Related Work

Pretrained Word Embeddings Many state-of-the-art models initialize their word representations using pretrained embeddings such as `word2vec` Mikolov et al. [2013a] or ELMo Peters et al. [2018]. These representations are typically trained using an interpretation of the Distributional Hypothesis Harris [1954] in which the bivariate distribution of target words and contexts is modeled. Our work deviates from the word embedding literature in two major aspects. First, our goal is to represent word *pairs*, not individual words. Second, our new PMI formulation models the *trivariate* word-word-context distribution. Experiments show that our pair embeddings can complement single-word embeddings.

Mining Textual Patterns There is extensive literature on mining textual patterns to predict relations between words Hearst [1992]; Snow et al. [2005]; Turney [2005]; Riedel et al. [2013]; Van de Cruys [2014]; Toutanova et al. [2015]; Shwartz and Dagan [2016]. These approaches focus mostly on relations between pairs of nouns (perhaps with the exception of VerbOcean Chklovski and Pantel [2004]). More recently, they have been expanded to predict relations between unrestricted pairs of words Jameel et al. [2018]; Espinosa Anke and Schockaert [2018], assuming that each word-pair was observed together during pretraining. Washio and Kato [2018a,b] relax this assumption with a compositional model that can represent any pair, as long as each word appeared (individually) in the corpus.

These methods are evaluated on either intrinsic relation prediction tasks, such as BLESS Baroni and Lenci [2011] and CogALex Santus et al. [2016], or knowledge-base population benchmarks, e.g. FB15 Bordes et al. [2013]. To the best of our knowledge, our work is the first to integrate pattern-based methods into modern high-performing semantic models and evaluate their impact on complex end-tasks like QA and NLI.

Integrating Knowledge in Complex Models Ahn et al. [2016] integrate Freebase facts into a language model using a copying mechanism over fact attributes. Yang and Mitchell [2017] modify the LSTM cell to incorporate WordNet and NELL knowledge for event and entity extraction. For cross-sentence inference tasks, Weissenborn et al. [2017], Bauer et al. [2018], and Mihaylov and Frank [2018] dynamically refine word representations by reading assertions from ConceptNet and Wikipedia abstracts. Our approach, on the

other hand, relies on a relatively simple extension of existing cross-sentence inference models. Furthermore, we do not need to dynamically retrieve and process knowledge base facts or Wikipedia texts, and just pretrain our pair vectors in advance.

KIM Chen et al. [2017] integrates word-pair vectors into the ESIM model for NLI in a very similar way to ours. However, KIM's word-pair vectors contain only hand-engineered word-relation indicators from WordNet, whereas our word-pair vectors are automatically learned from unlabeled text. Our vectors can therefore reflect relation types that do not exist in WordNet (such as *profession*) as well as word pairs that do not have a direct link in WordNet (e.g. *bronze* and *statue*); see Table 2.8 for additional examples.

2.6 Conclusion and Future Work

We presented new methods for training and using word *pair* embeddings that implicitly represent background knowledge. Our pair embeddings are computed as a compositional function of the individual word representations, which is learned by maximizing a variant of the PMI with the contexts in which the the two words co-occur. Experiments on cross-sentence inference benchmarks demonstrated that adding these representations to existing models results in sizable improvements for both in-domain and adversarial settings.

Published concurrently, BERT Devlin et al. [2018], which uses a masked language model objective, has reported dramatic gains on multiple semantic benchmarks including question-answering, natural language inference, and named entity recognition. Potential avenues for future work include multitasking BERT with `pair2vec` in order to more directly incorporate reasoning about word pair relations into the BERT objective.

Chapter 3

Improving Pre-training by Representing and Predicting Spans

This chapter discusses work originally published in Joshi et al. [2020a]

In this chapter, we focus on methods which encode knowledge beyond word pair relations into model parameters. Pre-training methods like BERT Devlin et al. [2018] have shown strong performance gains using self-supervised training that masks individual words or subword units. However, many NLP tasks involve reasoning about relationships between two or more spans of text. For example, in extractive question answering Rajpurkar et al. [2016], determining that the "Denver Broncos" is a type of "NFL team" is critical for answering the question "Which NFL team won Super Bowl 50?" Such spans provide a more challenging target for self supervision tasks, for example predicting "Denver Broncos" is much harder than predicting only "Denver" when you know the next word is "Broncos". In this paper, we introduce a span-level pre-training approach that consistently outperforms BERT, with the largest gains on span selection tasks such as question answering and coreference resolution.

We present SpanBERT, a pre-training method that is designed to better represent and predict spans of text. Our method differs from BERT in both the masking scheme and the training objectives. First, we mask random contiguous spans, rather than random individual tokens. Second, we introduce a novel *span-boundary objective* (SBO) so the model learns to predict the entire masked span from the observed tokens at its boundary. Span-based masking forces the model to predict entire spans solely using the context in

$$\mathcal{L}(\text{football}) = \mathcal{L}_{\text{MLM}}(\text{football}) + \mathcal{L}_{\text{SBO}}(\text{football})$$
$$= -\log P(\text{football} \mid \mathbf{x}_7) - \log P(\text{football} \mid \mathbf{x}_4, \mathbf{x}_9, \mathbf{p}_3)$$

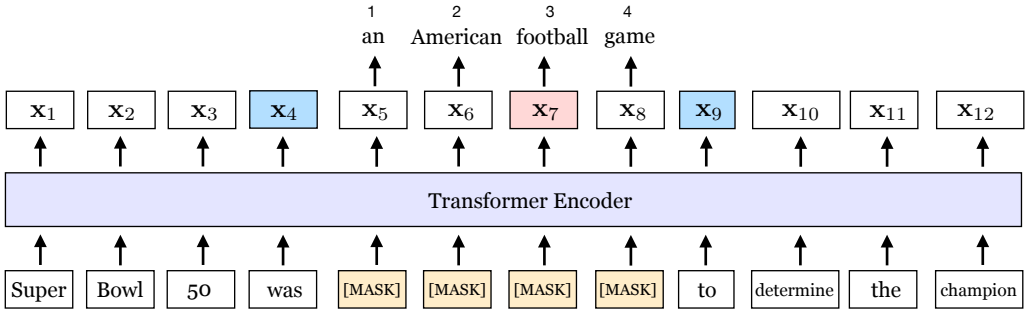

Figure 3.1: An illustration of SpanBERT training. The span *an American football game* is masked. The span boundary objective (SBO) uses the output representations of the boundary tokens, $\mathbf{x}_4$ and $\mathbf{x}_9$ (in blue), to predict each token in the masked span. The equation shows the MLM and SBO loss terms for predicting the token, *football* (in pink), which as marked by the position embedding $\mathbf{p}_3$, is the *third* token from x_4.

which they appear. Furthermore, the span-boundary objective encourages the model to store this span-level information at the boundary tokens, which can be easily accessed during the fine-tuning stage. Figure 3.1 illustrates our approach.

To implement SpanBERT, we build on a well-tuned replica of BERT, which itself substantially outperforms the original BERT. While building on our baseline, we find that pre-training on single segments, instead of two half-length segments with the next sentence prediction (NSP) objective, considerably improves performance on most downstream tasks. Therefore, we add our modifications on top of the tuned single-sequence BERT baseline.

Together, our pre-training process yields models that outperform all BERT baselines on a wide variety of tasks, and reach substantially better performance on span selection tasks in particular. Specifically, our method reaches 94.6% and 88.7% F1 on SQuAD 1.1 and 2.0 Rajpurkar et al. [2016, 2018], respectively — reducing error by as much as 27% compared to our tuned BERT replica. We also observe similar gains on five additional extractive question answering benchmarks (NewsQA, TriviaQA, SearchQA, HotpotQA, and Natural Questions).[1]

SpanBERT also arrives at a new state of the art on the challenging CoNLL-2012 ("OntoNotes") shared task for document-level coreference resolution, where we reach 79.6% F1, exceeding the previous top model

[1] We use the modified MRQA version of these datasets. See more details in Section 3.3.1.

by 6.6% absolute. Finally, we demonstrate that SpanBERT also helps on tasks that do not explicitly involve span selection, and show that our approach even improves performance on TACRED Zhang et al. [2017] and GLUE Wang et al. [2019].

While others show the benefits of adding more data Yang et al. [2019b] and increasing model size Lample and Conneau [2019], this work demonstrates the importance of designing good pre-training tasks and objectives, which can also have a remarkable impact.

3.1 Model

Our approach is inspired by BERT Devlin et al. [2018], but deviates from its bi-text classification framework in three ways. First, we use a different random process to mask *spans* of tokens, rather than individual ones. We also introduce a novel auxiliary objective – the span boundary objective (SBO) – which tries to predict the entire masked span using only the representations of the tokens at the span's boundary. Finally, SpanBERT samples a single contiguous segment of text for each training example (instead of two), and thus does not use BERT's next sentence prediction objective, which we omit.

3.1.1 Span Masking

Given a sequence of tokens $X = (x_1, x_2, \ldots, x_n)$, we select a subset of tokens $Y \subseteq X$ by iteratively sampling spans of text until the masking budget (e.g. 15% of X) has been spent. At each iteration, we first sample a span length (number of words) from a geometric distribution $\ell \sim \mathrm{Geo}(p)$, which is skewed towards shorter spans. We then randomly (uniformly) select the starting point for the span to be masked. We always sample a sequence of complete words (instead of subword tokens) and the starting point must be the beginning of one word.

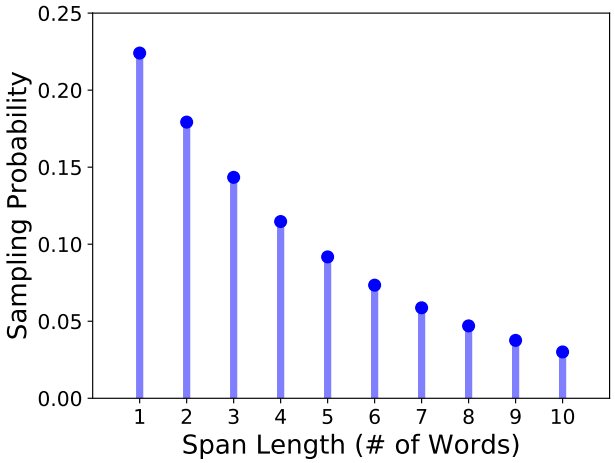

Figure 3.2: We sample random span lengths from a geometric distribution $\ell \sim \mathrm{Geo}(p = 0.2)$ clipped at $\ell_{max} = 10$.

Following preliminary trials[2], we set $p = 0.2$, and

also clip ℓ at $\ell_{max} = 10$. This yields a mean span length of $\mathrm{mean}(\ell) = 3.8$. Figure 3.2 shows the distribution of span mask lengths.

As in BERT, we also mask 15% of the tokens in total: replacing 80% of the masked tokens with `[MASK]`, 10% with random tokens and 10% with the original tokens. However, we perform this replacement at the span level and not for each token individually; i.e. all the tokens in a span are replaced with `[MASK]` or sampled tokens.

3.1.2 Span Boundary Objective (SBO)

Span selection models Lee et al. [2016, 2017]; He et al. [2018] typically create a fixed-length representation of a span using its boundary tokens (start and end). To support such models, we would ideally like the representations for the end of the span to summarize as much of the internal span content as possible. We do so by introducing a span boundary objective that involves predicting each token of a masked span using only the representations of the observed tokens at the boundaries (Figure 3.1).

Formally, we denote the output of the transformer encoder for each token in the sequence by $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Given a masked span of tokens $(x_s, \ldots, x_e) \in Y$, where (s, e) indicates its start and end positions, we represent each token x_i in the span using the output encodings of the *external* boundary tokens $\mathbf{x}_{s-1}$ and $\mathbf{x}_{e+1}$, as well as the position embedding of the target token $\mathbf{p}_{i-s+1}$:

$$\mathbf{y}_i = f(\mathbf{x}_{s-1}, \mathbf{x}_{e+1}, \mathbf{p}_{i-s+1})$$

where position embeddings $\mathbf{p}_1, \mathbf{p}_2, \ldots$ mark relative positions of the masked tokens with respect to the left boundary token x_{s-1}. We implement the representation function $f(\cdot)$ as a 2-layer feed-forward network with GeLU activations Hendrycks and Gimpel [2016] and layer normalization Ba et al. [2016]:

$$\begin{aligned}
\mathbf{h}_0 &= [\mathbf{x}_{s-1}; \mathbf{x}_{e+1}; \mathbf{p}_{i-s+1}] \\
\mathbf{h}_1 &= \mathrm{LayerNorm}(\mathrm{GeLU}(\mathbf{W}_1 \mathbf{h}_0)) \\
\mathbf{y}_i &= \mathrm{LayerNorm}(\mathrm{GeLU}(\mathbf{W}_2 \mathbf{h}_1))
\end{aligned}$$

[2]We experimented with $p = \{0.1, 0.2, 0.4\}$ and found 0.2 to perform the best.

We then use the vector representation $\mathbf{y}_i$ to predict the token x_i and compute the cross-entropy loss exactly like the MLM objective.

SpanBERT sums the loss from both the span boundary and the regular masked language model objectives for each token x_i in the masked span $(x_s, ..., x_e)$, while reusing the input embedding Press and Wolf [2017] for the target tokens in both MLM and SBO:

$$\begin{aligned}\mathcal{L}(x_i) &= \mathcal{L}_{\text{MLM}}(x_i) + \mathcal{L}_{\text{SBO}}(x_i) \\ &= -\log P\left(x_i \mid \mathbf{x}_i\right) - \log P\left(x_i \mid \mathbf{y}_i\right)\end{aligned}$$

3.2 Single-Sequence Training

BERT's examples contain two sequences of text (X_A, X_B), and an objective that trains the model to predict whether they are connected (NSP). We find that this setting is almost always worse than simply using a single sequence without the NSP objective (see Section 3.4 for further details). We conjecture that single-sequence training is superior to bi-sequence training with NSP because (a) the model benefits from longer full-length contexts, or (b) conditioning on, often unrelated, context from another document adds noise to the masked language model. Therefore, in our approach, we remove both the NSP objective and the two-segment sampling procedure, and simply sample a single contiguous segment of up to $n = 512$ tokens, rather than two half-segments that sum up to n tokens together.

In summary, SpanBERT pre-trains span representations by: (1) masking spans of full words using a geometric distribution based masking scheme (Section 3.1.1), (2) optimizing an auxiliary span-boundary objective (Section 3.1.2) in addition to MLM using a single-sequence data pipeline (Section 3.2).

3.3 Experimental Setup

3.3.1 Tasks

We evaluate on a comprehensive suite of tasks, including seven question answering tasks, coreference resolution, nine tasks in the GLUE benchmark Wang et al. [2019], and relation extraction. We expect that the span selection tasks, question answering and coreference resolution, will particularly benefit from our

span-based pre-training.

Extractive Question Answering Given a short passage of text and a question as input, the task of extractive question answering is to select a contiguous span of text in the passage as the answer.

We first evaluate on SQuAD 1.1 and 2.0 Rajpurkar et al. [2016, 2018], which have served as major question answering benchmarks, particularly for pre-trained models Peters et al. [2018]; Devlin et al. [2018]; Yang et al. [2019b]. We also evaluate on five more datasets from the MRQA shared task Fisch et al. [2019][3]: NewsQA Trischler et al. [2017], SearchQA Dunn et al. [2017], TriviaQA Joshi et al. [2017], HotpotQA Yang et al. [2018] and Natural Questions Kwiatkowski et al. [2019]. Because the MRQA shared task does not have a public test set, we split the development set in half to make new development and test sets. The datasets vary in both domain and collection methodology, making this collection a good testbed for evaluating whether our pre-trained models can generalize well across different data distributions.

Following BERT Devlin et al. [2018], we use the same QA model architecture for all the datasets. We first convert the passage $P = (p_1, p_2, \ldots, p_l)$ and question $Q = (q_1, q_2, \ldots, q_{l'})$ into a single sequence $X = \texttt{[CLS]}\, p_1 p_2 \ldots p_l\, \texttt{[SEP]}\, q_1 q_2 \ldots q_{l'}\, \texttt{[SEP]}$, pass it to the pre-trained transformer encoder, and train two linear classifiers independently on top of it for predicting the answer span boundary (start and end). For the unanswerable questions in SQuAD 2.0, we simply set the answer span to be the special token `[CLS]` for both training and testing.

Coreference Resolution Coreference resolution is the task of clustering mentions in text which refer to the same real-world entities. We evaluate on the CoNLL-2012 shared task Pradhan et al. [2012] for document-level coreference resolution. We use the *independent* version of the Joshi et al. [2019b] implementation of the higher-order coreference model Lee et al. [2018]. The document is divided into non-overlapping segments of a pre-defined length.[4] Each segment is encoded independently by the pre-trained transformer encoder, which replaces the original LSTM-based encoder. For each mention span x, the model learns a distribution $P(\cdot)$ over possible antecedent spans Y:

[3]https://github.com/mrqa/MRQA-Shared-Task-2019. MRQA changed the original datasets to unify them into the same format, e.g. all the contexts are truncated to a maximum of 800 tokens and only answerable questions are kept.

[4]The length was chosen from {128, 256, 384, 512}.

$$P(y) = \frac{e^{s(x,y)}}{\sum_{y' \in Y} e^{s(x,y')}}$$

The span pair scoring function $s(x, y)$ is a feedforward neural network over fixed-length span representations and hand-engineered features over x and y:

$$s(x, y) = s_m(x) + s_m(y) + s_c(x, y)$$
$$s_m(x) = \text{FFNN}_m(\mathbf{g_x})$$
$$s_c(x, y) = \text{FFNN}_c(\mathbf{g_x}, \mathbf{g_y}, \phi(x, y))$$

Here $\mathbf{g_x}$ and $\mathbf{g_y}$ denote the span representations, which are a concatenation of the two transformer output states of the span endpoints and an attention vector computed over the output representations of the token in the span. FFNN_m and FFNN_c represent two feedforward neural networks with one hidden layer, and $\phi(x, y)$ represents the hand-engineered features (e.g. speaker and genre information). A more detailed description of the model can be found in Joshi et al. [2019b].

Relation Extraction TACRED Zhang et al. [2017] is a challenging relation extraction dataset. Given one sentence and two spans within it – subject and object – the task is to predict the relation between the spans from 42 pre-defined relation types, including *no_relation*. We follow the entity masking schema from Zhang et al. [2017] and replace the subject and object entities by their NER tags such as "`[CLS] [SUBJ-PER]` was born in `[OBJ-LOC]` , Michigan, ...", and finally add a linear classifier on top of the `[CLS]` token to predict the relation type.

GLUE The General Language Understanding Evaluation (GLUE) benchmark Wang et al. [2019] consists of 9 sentence-level classification tasks:

- Two *sentence-level classification* tasks including CoLA Warstadt et al. [2018] for evaluating linguistic acceptability and SST-2 Socher et al. [2013] for sentiment classification.
- Three *sentence-pair similarity* tasks including MRPC Dolan and Brockett [2005], a binary paraphrasing task sentence pairs from news sources, STS-B Cer et al. [2017], a graded similarity task for news

headlines, and QQP[5], a binary paraphrasing tasking between Quora question pairs.

- Four *natural language inference* tasks including MNLI Williams et al. [2018b], QNLI Rajpurkar et al. [2016], RTE Dagan et al. [2005]; Bar-Haim et al. [2006]; Giampiccolo et al. [2007] and WNLI Levesque et al. [2011].

Unlike question answering, coreference resolution, and relation extraction, these sentence-level tasks do not require *explicit* modeling of span-level semantics. However, they might still benefit from implicit span-based reasoning (e.g., *the Prime Minister* is *the head of the government*). Following previous work Devlin et al. [2018]; Radford et al. [2018][6], we exclude WNLI from the results to enable a fair comparison. While recent work Liu et al. [2019b] has applied several task-specific strategies to increase performance on the individual GLUE tasks, we follow BERT's single-task setting and only add a linear classifier on top of the `[CLS]` token for these classification tasks.

3.3.2 Implementation

We reimplemented BERT's model and pre-training method in *fairseq* Ott et al. [2019]. We used the model configuration of $\text{BERT}_{\text{large}}$ as in 352018Devlin et al.Devlin, Chang, Lee, and Toutanova) and also pre-trained all our models on the same corpus: BooksCorpus and English Wikipedia using *cased* Wordpiece tokens.

Compared to the original BERT implementation, the main differences in our implementation include: (a) We use different masks at each epoch while BERT samples 10 different masks for each sequence during data processing. (b) We remove all the short-sequence strategies used before (they sampled shorter sequences with a small probability 0.1; they also first pre-trained with smaller sequence length of 128 for 90% of the steps). Instead, we always take sequences of up to 512 tokens until it reaches a document boundary. We refer readers to 912019cLiu et al.Liu, Ott, Goyal, Du, Joshi, Chen, Levy, Lewis, Zettlemoyer, and Stoyanov) for further discussion on these modifications and their effects.

As in BERT, the learning rate is warmed up over the first 10,000 steps to a peak value of 1e-4, and then linearly decayed. We retain β hyperparameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) and a decoupled weight decay Loshchilov and Hutter [2019] of 0.1. We also keep a dropout of 0.1 on all layers and attention weights, and

[5] https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

[6] Previous work has excluded WNLI on account of construction issues outlined on the GLUE website – `https://gluebenchmark.com/faq`

a GeLU activation function Hendrycks and Gimpel [2016]. We deviate from the optimization by running for 2.4M steps and using an epsilon of 1e-8 for AdamW Kingma and Ba [2015], which converges to a better set of model parameters. Our implementation uses a batch size of 256 sequences with a maximum of 512 tokens.[7] For the SBO, we use 200 dimension position embeddings $\mathbf{p}_1, \mathbf{p}_2, \ldots$ to mark positions relative to the left boundary token. The pre-training was done on 32 Volta V100 GPUs and took 15 days to complete.

Fine-tuning is implemented based on HuggingFace's codebase Wolf et al. [2019]. We refer the reader to Joshi et al. [2020a] for further details.

3.3.3 Baselines

We compare SpanBERT to three baselines:

Google BERT The pre-trained models released by 352018Devlin et al.Devlin, Chang, Lee, and Toutanova).[8]

Our BERT Our reimplementation of BERT with improved data preprocessing and optimization (subsection 3.3.2).

Our BERT-1seq Our reimplementation of BERT trained on single full-length sequences without NSP (subsection 3.2).

3.4 Results

We compare SpanBERT to the baselines per task, and draw conclusions based on the overall trends.

3.4.1 Per-Task Results

Extractive Question Answering Table 3.1 shows the performance on both SQuAD 1.1 and 2.0. SpanBERT exceeds our BERT baseline by 2.0% and 2.8% F1 respectively (3.3% and 5.4% over Google BERT). In SQuAD 1.1, this result accounts for over 27% error reduction, reaching 3.4% F1 *above* human performance.

[7] On the average, this is approximately 390 sequences since some documents have fewer than 512 tokens

[8] https://github.com/google-research/bert.

	SQuAD 1.1		SQuAD 2.0	
	EM	F1	EM	F1
Human Perf.	82.3	91.2	86.8	89.4
Google BERT	84.3	91.3	80.0	83.3
Our BERT	86.5	92.6	82.8	85.9
Our BERT-1seq	87.5	93.3	83.8	86.6
SpanBERT	**88.8**	**94.6**	**85.7**	**88.7**

t

Table 3.1: Test results on SQuAD 1.1 and SQuAD 2.0.

	NewsQA	TriviaQA	SearchQA	HotpotQA	Natural Questions	Avg.
Google BERT	68.8	77.5	81.7	78.3	79.9	77.3
Our BERT	71.0	79.0	81.8	80.5	80.5	78.6
Our BERT-1seq	71.9	80.4	84.0	80.3	81.8	79.7
SpanBERT	**73.6**	**83.6**	**84.8**	**83.0**	**82.5**	**81.5**

Table 3.2: Performance (F1) on the five MRQA extractive question answering tasks.

Table 3.2 demonstrates that this trend goes beyond SQuAD, and is consistent in every MRQA dataset. On average, we see a 2.9% F1 improvement from our reimplementation of BERT. Although some gains are coming from single-sequence training (+1.1%), most of the improvement stems from span masking and the span boundary objective (+1.8%), with particularly large gains on TriviaQA (+3.2%) and HotpotQA (+2.7%).

	MUC			B^3			$CEAF_{\phi_4}$			
	P	R	F1	P	R	F1	P	R	F1	Avg. F1
Prev. SotA: Lee et al. [2018]	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	73.0
Google BERT	84.9	82.5	83.7	76.7	74.2	75.4	74.6	70.1	72.3	77.1
Our BERT	85.1	83.5	84.3	77.3	75.5	76.4	75.0	71.9	73.9	78.3
Our BERT-1seq	85.5	84.1	84.8	77.8	76.7	77.2	75.3	73.5	74.4	78.8
SpanBERT	**85.8**	**84.8**	**85.3**	**78.3**	**77.9**	**78.1**	**76.4**	**74.2**	**75.3**	**79.6**

Table 3.3: Performance on the OntoNotes coreference resolution benchmark. The main evaluation is the average F1 of three metrics: MUC, B^3, and $CEAF_{\phi_4}$ on the test set.

Coreference Resolution Table 3.3 shows the performance on the OntoNotes coreference resolution benchmark. Our BERT reimplementation improves the Google BERT model by 1.2% on the average F1 metric and single-sequence training brings another 0.5% gain. Finally, SpanBERT improves considerably on top of that, achieving a new state of the art of 79.6% F1 (previous best result is 73.0%).

		P	R	F1
t	BERT_{EM} Soares et al. [2019]	-	-	70.1
	BERT_{EM}+MTB*	-	-	**71.5**
	Google BERT	69.1	63.9	66.4
	Our BERT	67.8	67.2	67.5
	Our BERT-1seq	**72.4**	67.9	70.1
	SpanBERT	70.8	**70.9**	**70.8**

Table 3.4: Test performance on the TACRED relation extraction benchmark.

	CoLA	SST-2	MRPC	STS-B	QQP	MNLI	QNLI	RTE	(Avg)
Google BERT	59.3	**95.2**	88.5/84.3	86.4/88.0	71.2/89.0	86.1/85.7	93.0	71.1	80.4
Our BERT	58.6	93.9	90.1/86.6	88.4/89.1	71.8/89.3	87.2/86.6	93.0	74.7	81.1
Our BERT-1seq	63.5	94.8	**91.2**/87.8	89.0/88.4	**72.1/89.5**	88.0/87.4	93.0	72.1	81.7
SpanBERT	**64.3**	94.8	90.9/**87.9**	**89.9/89.1**	71.9/**89.5**	**88.1/87.7**	**94.3**	**79.0**	**82.8**

Table 3.5: Test set performance on GLUE tasks. MRPC: F1/accuracy, STS-B: Pearson/Spearmanr correlation, QQP: F1/accuracy, MNLI: matched/mistached accuracies and accuracy for all the other tasks. WNLI (not shown) is always set to majority class (65.1% accuracy) and included in the average.

Relation Extraction Table 3.4 shows the performance on TACRED. SpanBERT exceeds our reimplementation of BERT by 3.3% F1 and achieves close to the current state of the art Soares et al. [2019] — Our model performs better than their BERT_{EM} but is 0.7 point behind BERT_{EM} + MTB which used entity-linked text for additional pre-training. Most of this gain (+2.6%) stems from single-sequence training although the contribution of span masking and the span boundary objective is still a considerable 0.7%, resulting largely from higher recall.

GLUE Table 3.5 shows the performance on GLUE. For most tasks, the different models appear to perform similarly. Moving to single-sequence training without the NSP objective substantially improves CoLA, and yields smaller (but considerable) improvements on MRPC and MNLI. The main gains from SpanBERT are in the SQuAD-based QNLI dataset (+1.3%) and in RTE (+6.9%), the latter accounting for most of the rise in SpanBERT's GLUE average.

3.4.2 Overall Trends

We compared our approach to three BERT baselines on 17 benchmarks, and found that **SpanBERT outperforms BERT on almost every task**. In 14 tasks, SpanBERT performed better than all baselines. In 2

	SQuAD 2.0	NewsQA	TriviaQA	Coreference	MNLI-m	QNLI	GLUE (Avg)
Subword Tokens	83.8	72.0	76.3	**77.7**	86.7	92.5	83.2
Whole Words	84.3	72.8	77.1	76.6	86.3	92.8	82.9
Named Entities	84.8	72.7	78.7	75.6	86.0	93.1	83.2
Noun Phrases	85.0	**73.0**	77.7	76.7	86.5	93.2	**83.5**
Geometric Spans	**85.4**	**73.0**	**78.8**	76.4	**87.0**	**93.3**	83.4

Table 3.6: The effect of replacing BERT's original masking scheme (Subword Tokens) with different masking schemes. Results are F1 scores for QA tasks and accuracy for MNLI and QNLI on the development sets. All the models are based on bi-sequence training with NSP.

	SQuAD 2.0	NewsQA	TriviaQA	Coref	MNLI-m	QNLI	GLUE (Avg)
Span Masking (2seq) + NSP	85.4	73.0	78.8	76.4	87.0	93.3	83.4
Span Masking (1seq)	86.7	73.4	80.0	76.3	87.3	93.8	83.8
Span Masking (1seq) + SBO	**86.8**	**74.1**	**80.3**	**79.0**	**87.6**	**93.9**	**84.0**

Table 3.7: The effects of different auxiliary objectives, given MLM over random spans as the primary objective.

tasks (MRPC and QQP), it performed on-par in terms of accuracy with single-sequence trained BERT, but still outperformed the other baselines. In one task (SST-2), Google's BERT baseline performed better than SpanBERT by 0.4% accuracy.

When considering the magnitude of the gains, it appears that **SpanBERT is especially better at extractive question answering**. In SQuAD 1.1, for example, we observe a solid gain of 2.0% F1 even though the baseline is already well above human performance. On MRQA, SpanBERT improves between 2.0% (Natural Questions) and 4.6% (TriviaQA) F1 on top of our BERT baseline.

Finally, we observe that **single-sequence training works considerably better than bi-sequence training with next sentence prediction (NSP)** with BERT's choice of sequence lengths for a wide variety of tasks . This is surprising because BERT's ablations showed gains from the NSP objective Devlin et al. [2018]. However, the ablation studies still involved bi-sequence data processing, i.e. the pre-training stage only controlled for the NSP objective while still sampling two half-length sequences. We hypothesize that bi-sequence training, as it is implemented in BERT, impedes the model from learning longer-range features, and consequently hurts performance on many downstream tasks.

3.5 Ablation Studies

We compare our random span masking scheme with linguistically-informed masking schemes, and find that masking random spans is a competitive and often better approach. We then study the impact of the span boundary objective (SBO), and contrast it with BERT's next sentence prediction (NSP) objective.[9]

3.5.1 Masking Schemes

Previous work Sun et al. [2019b] has shown improvements in downstream task performance by masking linguistically-informed spans during pre-training for Chinese data. We compare our random span masking scheme with masking of linguistically-informed spans. Specifically, we train the following five baseline models differing only in the way tokens are masked.

Subword Tokens We sample random Wordpiece tokens, as in the original BERT.

Whole Words We sample random words, and then mask all of the subword tokens in those words. The total number of masked subtokens is around 15%.

Named Entities At 50% of the time, we sample from named entities in the text, and sample random whole words for the other 50%. The total number of masked subtokens is 15%. Specifically, we run spaCy's named entity recognizer[10] on the corpus and select all the non-numerical named entity mentions as candidates.

Noun Phrases Similar as *Named Entities*, we sample from noun phrases at 50% of the time. The noun phrases are extracted by running spaCy's constituency parser.

Geometric Spans We sample random spans from a geometric distribution, as in our SpanBERT (see Section 3.1.1).

Table 3.6 shows how different pre-training masking schemes affect performance on the development set of a selection of tasks. All the models are evaluated on the development sets and are based on the default BERT setup of bi-sequence training with NSP; the results are not directly comparable to the main

[9] To save time and resources, we use the checkpoints at 1.2M steps for all the ablation experiments.

[10] `https://spacy.io/`

evaluation. With the exception of coreference resolution, masking random spans is preferable to other strategies. Although linguistic masking schemes (named entities and noun phrases) are often competitive with random spans, their performance is not consistent; for instance, masking noun phrases achieves parity with random spans on NewsQA, but underperforms on TriviaQA (-1.1% F1).

On coreference resolution, we see that masking random subword tokens is preferable to any form of span masking. Nevertheless, we shall see in the following experiment that combining random span masking with the span boundary objective can improve upon this result considerably.

3.5.2 Auxiliary Objectives

In Section 3.4, we saw that bi-sequence training with the next sentence prediction (NSP) objective can hurt performance on downstream tasks, when compared to single-sequence training. We test whether this holds true for models pre-trained with span masking, and also evaluate the effect of replacing the NSP objective with the span boundary objective (SBO).

Table 3.7 confirms that single-sequence training typically improves performance. Adding SBO further improves performance, with a substantial gain on coreference resolution (+2.7% F1) over span masking alone. Unlike the NSP objective, SBO does not appear to have any adverse effects.

3.6 Related Work

Pre-trained contextualized word representations that can be trained from unlabeled text Dai and Le [2015]; Melamud et al. [2016]; Peters et al. [2018] have had immense impact on NLP lately, particularly as methods for initializing a large model before fine-tuning it for a specific task Howard and Ruder [2018]; Radford et al. [2018]; Devlin et al. [2018]. Beyond differences in model hyperparameters and corpora, these methods mainly differ in their pre-training tasks and loss functions, with a considerable amount of contemporary literature proposing augmentations of BERT's masked language modeling (MLM) objective.

While previous and concurrent work has looked at masking Sun et al. [2019b] or dropping Song et al. [2019]; Chan et al. [2019] multiple words from the input – particularly as pretraining for language generation tasks – SpanBERT pretrains span representations Lee et al. [2016], which are widely used for question answering, coreference resolution and a variety of other tasks. ERNIE Sun et al. [2019b]

shows improvements on Chinese NLP tasks using phrase and named entity masking. MASS Song et al. [2019] focuses on language generation tasks, and adopts the encoder-decoder framework to reconstruct a sentence fragment given the remaining part of the sentence. We attempt to more explicitly model spans using the SBO objective, and show that (geometrically distributed) random span masking works as well, and sometimes better than, masking linguistically-coherent spans. We evaluate on English benchmarks for question answering, relation extraction, and coreference resolution in addition to GLUE.

A different ERNIE Zhang et al. [2019b] focuses on integrating structured knowledge bases with contextualized representations with an eye on knowledge-driven tasks like entity typing and relation classification. UNILM Dong et al. [2019] uses multiple language modeling objectives – unidirectional (both left-to-right and right-to-left), bidirectional, and sequence-to-sequence prediction – to aid generation tasks like summarization and question generation. XLM Lample and Conneau [2019] explores cross-lingual pre-training for multilingual tasks such as translation and cross-lingual classification. Kermit Chan et al. [2019], an insertion based approach, fills in missing tokens (instead of predicting masked ones) during pretraining; they show improvements on machine translation and zero-shot question answering.

Concurrent with our work, RoBERTa Liu et al. [2019c] presents a replication study of BERT pre-training that measures the impact of many key hyperparameters and training data size. Also concurrent, XLNet Yang et al. [2019b] combines an autoregressive loss and the Transformer-XL Dai et al. [2019] architecture with a more than an eight-fold increase in data to achieve current state-of-the-art results on multiple benchmarks. XLNet also masks spans (of 1-5 tokens) during pre-training, but predicts them autoregressively. Our model focuses on incorporating span-based pre-training, and as a side effect, we present a stronger BERT baseline while controlling for the corpus, architecture, and the number of parameters.

Related to our SBO objective, *pair2vec* Joshi et al. [2019a] encodes word-pair relations using a negative sampling-based multivariate objective during pre-training. Later, the word-pair representations are injected into the attention-layer of downstream tasks, and thus encode limited downstream context. Unlike pair2vec, our SBO objective yields "pair" (start and end tokens of spans) representations which more fully encode the context during both pre-training and finetuning, and are thus more appropriately viewed as *span* representations. Stern et al. [2018] focus on improving language generation speed using a block-wise parallel decoding scheme; they make predictions for multiple time steps in parallel and then back off to the longest

prefix validated by a scoring model. Also related are sentence representation methods Kiros et al. [2015]; Logeswaran and Lee [2018] which focus on predicting surrounding contexts from sentence embeddings.

3.7 Conclusion

We presented a new method for span-based pre-training which extends BERT by (1) masking contiguous random spans, rather than random tokens, and (2) training the span boundary representations to predict the entire content of the masked span, without relying on the individual token representations within it. Together, our pre-training process yields models that outperform all BERT baselines on a variety of tasks, and reach substantially better performance on span selection tasks in particular.

Chapter 4

Representations Using Dynamically Retrieved Textual Knowledge

This chapter discusses work originally published in Joshi et al. [2020b]

Current self-supervised representations, trained at large scale from document-level contexts, are known to encode linguistic Tenney et al. [2019] and factual Petroni et al. [2019] knowledge into their parameters. Yet, even large pretrained representations are unable to capture and preserve all factual knowledge they have "read" during pretraining due to the long tail of entity and event-specific information Logan et al. [2019].

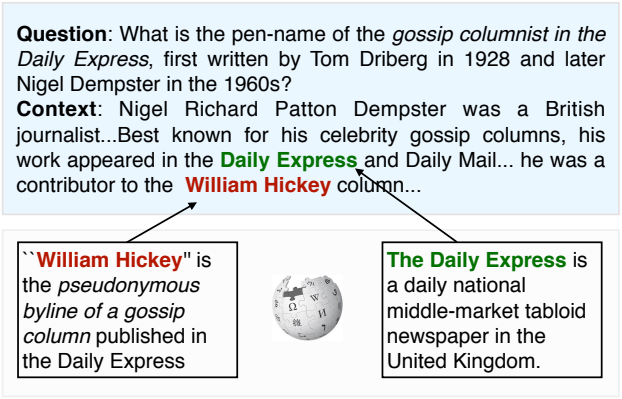

Figure 4.1: A TriviaQA example showing how background sentences from Wikipedia help define the meaning of phrases in the context and their relationship to phrases in the question. The answer *William Hickey* is connected to the question phrase *pen-name of the gossip columnist in the Daily Express* through the background sentence.

On the other hand, when relevant text is provided as input, such as in reading comprehension tasks Rajpurkar et al. [2016], relation extraction, syntactic analysis, etc., which can be cast as tasks of labeling spans in the input text, prior work has not focused on drawing background information from external text sources. Instead, most research has

explored architectures to integrate background from structured knowledge bases to form input text representations Bauer et al. [2018]; Mihaylov and Frank [2018]; Yang et al. [2019a]; Zhang et al. [2019b]; Peters et al. [2019].[1]

We posit that representations should be able to directly integrate *textual* background knowledge since a wider scope of information is more readily available in textual form. Our method represents input texts by jointly encoding them with *dynamically retrieved* sentences from the Wikipedia pages of entities they mention. We term these representations TEK-enriched, for Textual Encyclopedic Knowledge (Figure 4.2 shows an illustration), and use them for reading comprehension (RC) by contextualizing questions and passages together with retrieved Wikipedia background sentences. Such background knowledge can help reason about the relationships between questions and passages. Figure 4.1 shows an example question from the TriviaQA dataset Joshi et al. [2017] asking for *the pen-name of a gossip columnist*. Encoding relevant background knowledge (*pseudonymous byline of a gossip column published in the Daily Express*) helps ground the vague reference to *the William Hickey column* in the given document context.

Using text as background knowledge allows us to directly reuse powerful pretrained BERT-style encoders Devlin et al. [2018]. We show that an off-the-shelf RoBERTa Liu et al. [2019c] model can be directly finetuned on minimally structured TEK-enriched inputs, which are formatted to allow the encoder to distinguish between the original passages and background sentences. This method considerably improves on current state-of-the art methods which only consider context from a single input document (Section 3.4). The improvement comes without an increase in the length of the input window for the Transformer Vaswani et al. [2017].

Although existing pretrained models provide a good starting point for task-specific TEK-enriched representations, there is still a mismatch between the type of input seen during pretraining (single document segments) and the type of input the model is asked to represent for downstream tasks (document text with background Wikipedia sentences from multiple pages). We show that the Transformer model can be substantially improved by reducing this mismatch via self-supervised masked language model (MLM) Devlin et al. [2018] pretraining on TEK-augmented input texts.

Our approach records considerable improvements over state of the art base (12-layer) and large (24-

[1] A notable exception is 1592017Weissenborn et al.Weissenborn, Kočiskỳ, and Dyer), with a specialized architecture which uses *textual* entity descriptions.

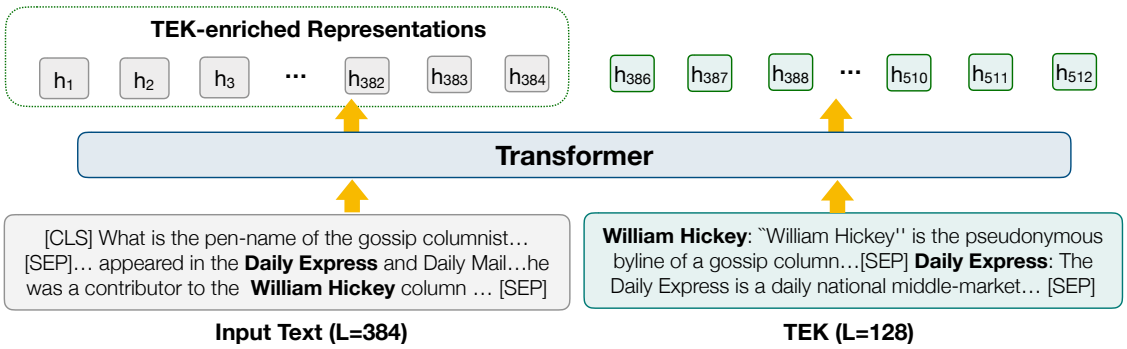

Figure 4.2: We contextualize the input text, in this case a question and a passage, together with textual encyclopedic knowledge (TEK) using a pretrained Transformer to create TEK-enriched representations.

layer) Transformer models for in-domain and out-of-domain document-level extractive question answering (QA), for tasks where factual knowledge about entities is important and well-covered by the background collection. On TriviaQA, we see improvements of 1.6 to 3.1 F1, respectively, over comparable RoBERTa models which do not integrate background information. On MRQA Fisch et al. [2019], a large collection of diverse QA datasets, we see consistent gains in-domain along with large improvements out-of-domain on BioASQ (2.1 to 4.2 F1), TextbookQA (1.6 to 2.0 F1), and DuoRC (1.1 to 2.0 F1).

4.1 TEK-Enriched Representations

We follow recent work on pretraining bidirectional Transformer representations on unlabeled text, and fine-tuning them for downstream tasks Devlin et al. [2018]. Subsequent approaches have shown significant improvements over BERT by improving the training example generation, the masking strategy, the pretraining objectives, and the optimization methods Liu et al. [2019c]; Joshi et al. [2020a]. We build on these improvements to train TEK-enriched representations and use them for extractive QA.

Our approach seeks to contextualize input text $X = (x_1, \ldots, x_n)$ jointly with relevant textual encyclopedic background knowledge B retrieved dynamically from multiple documents. We define a retrieval function, $f_{ret}(X, \mathcal{D})$, which takes X as input and retrieves a list of text spans $B = (B_1, \ldots, B_M)$ from the corpus $\mathcal{D}$. In our implementation, each of the text spans B_i is a sentence. The encoder then represents X by jointly encoding X with B using $f_{enc}(X, B)$ such that the output representations of X are cognizant of the information present in B (see Figure 4.2). We use a deep Transformer encoder operating over the input

sequence [CLS] X [SEP] B [SEP] for $f_{enc}(\cdot)$.

We refer to inputs X generically as *contexts*. These could be either contiguous word sequences from documents (passages), or, for the QA application, question-passage pairs, which we refer to as *RC-contexts*. For a fixed Transformer input length limit (which is necessary for computational efficiency), there is a trade-off between the length of the document context (the length of X) and the amount of background knowledge (the length of B). subsection 4.4 explores this trade-off in detail with experiments showing that for an encoder input limit of 512, the values of $N_C = 384$ for the length of X and $N_B = 128$ for the length of B provide an effective compromise.

We use a simple implementation of the background retrieval function $f_{ret}(X, \mathcal{D})$, using an entity linker for finetuning (subsection 4.1.1) and Wikipedia hyperlinks for pretraining (subsection 4.1.2), and a way to score the relevance of individual sentences using *ngram* overlap.

4.1.1 TEK-Enriched Question Answering

The input X for the extractive QA task consists of the question Q and a candidate passage P. We use the following retrieval function $f_{ret}(X)$ to obtain relevant background B.

Background Knowledge Retrieval for QA We detect entity mentions in X using a proprietary Wikipedia-based entity linker,[2] and form a candidate pool of background segments B_i as the union of the sentences in the Wikipedia pages of the detected entities. These sentences are then ranked based on their number of overlapping ngrams with the question (equally weighted unigrams, bigrams, and trigrams). To form the input for the Transformer encoder, each background sentence is minimally structured as B_i by prepending the name of the entity whose page it belongs to along with a separator ':' token. Each sentence B_i is followed by [SEP].

QA Model Following BERT, our QA model architecture consists of two independent linear classifiers for predicting the answer span boundary (start and end) on top of the output representations of X. We assume that the answer, if present, is contained only in the given passage, P, and do not consider potential mentions

[2]We also report results on publicly available linkers showing that our method is robust to the exact choice of the linker (subsection 4.4).

of the answer in the background B. For instances which do not contain the answer, we set the answer span to be the special token `[CLS]`. We use a fixed Transformer input window size of 512, and use a sliding window with a stride of 128 tokens to handle longer documents. Our TEK-enriched representations use document passages of length 384 while baselines use longer passages of length 512.

4.1.2 TEK-Enriched Pretraining

Standard pretraining uses contiguous document-level natural language inputs. Since TEK-augmented inputs are formatted as natural language sequences, off-the-shelf pretrained models can be used as a starting point for creating TEK-enriched representations. As one of our approaches, we use a standard single-document pretraining model.

While the input format is the same, there is a mismatch between contiguous document segments and TEK-augmented inputs sourced from multiple documents. We propose an additional pretraining stage—starting from the RoBERTa parameters, we resume pretraining using an MLM objective on TEK-augmented document text X, which encourages the model to integrate the knowledge from multiple background segments.

Background Knowledge Retrieval in Pretraining In pretraining, X is a contiguous block of text from Wikipedia. The retrieval function $f_{ret}(X, \mathcal{D})$ returns $B = (B_1, \ldots, B_M)$ where each B_i is a sentence from the Wikipedia page of some entity hyperlinked from a span in X. We use high-precision Wikipedia hyperlinks instead of an entity linker for pretraining. The background candidate sentences are ranked by their ngram overlap with X. The top ranking sentences in B up to N_B tokens are used. If no entities are found in X, B is constructed from the context following X from the same document.

Training Objective We continue pretraining a deep Transformer using the MLM objective Devlin et al. [2018] after initializing the parameters with pretrained RoBERTa weights. Following improvements in SpanBERT Joshi et al. [2020a], we mask spans with lengths sampled from a geometric distribution in the entire input (X and B). We use a single segment ID, and remove the next sentence prediction objective which has been shown to not improve performance Joshi et al. [2020a]; Liu et al. [2019c] for multiple tasks including QA.

Task	Train	Dev	Test
t TQA Wiki	61,888	7,993	7,701
TQA Web	528,979	68,621	65,059
MRQA	616,819	58,221	9,633

Table 4.1: Data statistics for TriviaQA and MRQA.

We evaluate two methods building textual-knowledge enriched representations for QA differing in the pretraining approach used:

TEK$_{PF}$ Our full approach TEK$_{PF}$ consists of two stages: (a) 200K steps of TEK-pretraining on Wikipedia starting from the RoBERTa checkpoint, and (b) finetuning and doing inference on RC-contexts augmented with TEK background.

TEK$_F$ TEK$_F$ replaces the first specialized pretraining stage in TEK$_{PF}$ with 200K steps for standard single-document-context pretraining for a fair comparison with TEK$_{PF}$, but follows the same finetuning regimen.

The subscripts P and F stand for pretraining and finetuning, respectively.

4.2 Experimental Setup

We perform experiments on TriviaQA and MRQA, two large extractive question answering benchmarks (see Table 4.1 for dataset statistics).

TriviaQA TriviaQA Joshi et al. [2017] contains trivia questions paired with evidence collected via entity linking and web search. The dataset is *distantly supervised* in that the answers are contained in the evidence but the context may not support answering the questions. We experiment with both the Wikipedia and Web tasks. For further details on dataset processing, we refer the reader to Joshi et al. [2020b]

MRQA The MRQA shared task Fisch et al. [2019] consists of several widely used QA datasets unified into a common format aimed at evaluating out-of-domain generalization. The data consists of a training set, in-domain and out-of-domain dev sets, and a private out-of-domain test set. The training and the in-

domain dev sets consist of modified versions of corresponding sets from SQuAD Rajpurkar et al. [2016], NewsQA Trischler et al. [2017], SearchQA Dunn et al. [2017], TriviaQA Web Joshi et al. [2017], HotpotQA Yang et al. [2018] and Natural Questions Kwiatkowski et al. [2019]. The out-of-domain test evaluation, including access to questions and passages, is only available through Codalab. Due to the complexity of our system which involves entity linking and retrieval, we perform development and model selection on the in-domain dev set and treat the out-of-domain dev set as the test set. The out-of-domain set we evaluate on has examples from BioASQ Tsatsaronis et al. [2015], DROP Dua et al. [2019], DuoRC Saha et al. [2018], RACE Lai et al. [2017], RelationExtraction Levy et al. [2017], and TextbookQA Kembhavi et al. [2017].

4.2.1 Baselines

We compare TEK$_{PF}$ and TEK$_F$ with two baselines, RoBERTa and RoBERTa++. Both use the same architecture as our approach, but use only original RC-contexts for finetuning and inference, and use standard single-document-context RoBERTa pretraining. TEK$_{PF}$ and TEK$_F$ use $N_C = 384$ and $N_B = 128$, while both baselines use $N_C = 512$ and $N_B = 0$.

RoBERTa We finetune the model on QA data without knowledge augmentation starting from the same RoBERTa checkpoint that is used as an initializer for TEK-augmented pretraining.

RoBERTa++ For a fair evaluation of the new TEK-augmented pretraining method while controlling for the number of pretraining steps and other hyperparameters, we extend RoBERTa's pretraining for an additional 200K steps on single contiguous blocks of text (without background information). We use the same masking and other hyperparameters as in TEK-augmented pretraining. This pretrained checkpoint is also used to initialize parameters for our TEK$_F$ approach.

4.3 Results

TriviaQA Table 4.2 compares our approaches with baselines and previous work. The 12-layer variant of our RoBERTa baseline outperforms or matches the performance of several previous systems including ELMo-based ones Wang et al. [2018]; Lewis [2018] which are specialized for this task. We also see that

	TQA Wiki		TQA Web	
	EM	F1	EM	F1
Previous work				
Clark and Gardner [2018]	64.0	68.9	66.4	71.3
Weissenborn et al. [2017]	64.6	69.9	67.5	72.8
Wang et al. [2018]	66.6	71.4	68.6	73.1
Lewis [2018]	67.3	72.3	-	-
!t This work				
RoBERTa (Base)	66.7	71.7	77.0	81.4
RoBERTa++ (Base)	68.0	72.9	76.8	81.4
TEK$_F$ (Base)	70.0	74.8	78.2	83.0
TEK$_{PF}$ (Base)	**71.2**	**76.0**	**78.8**	**83.4**
RoBERTa (Large)	72.3	76.9	80.6	85.1
RoBERTa++ (Large)	72.9	77.5	81.1	85.5
TEK$_F$(Large)	74.1	78.6	82.2	86.5
TEK$_{PF}$ (Large)	**74.6**	**79.1**	**83.0**	**87.2**

Table 4.2: Test set performance on TriviaQA.

RoBERTa++ outperforms RoBERTa, indicating that despite large scale pretraining, there is still room for improvement by simply pretraining for more steps on task-domain relevant text.

Furthermore, the 12-layer and 24-layer variants of our TEK$_F$ approach considerably improve over a comparable RoBERTa++ baseline for both Wikipedia (1.9 and 1.1 F1 respectively) and Web (1.6 and 1.0 F1 respectively) indicating that TEK representations are useful even without additional TEK-pretraining. The base variant of our best model TEK$_{PF}$, which uses TEK-pretrained TEK-enriched representations records even bigger gains of 3.1 F1 and 2.0 F1 on Wikipedia and Web respectively over a comparable 12-layer RoBERTa++ baseline. The 24-layer models show similar trends with improvements of 1.6 F1 and 1.7 F1 over RoBERTa++.

MRQA Table 4.3 shows in-domain and out-of-domain evaluation on MRQA. As in the case of TriviaQA, the 12-layer variants of our RoBERTa baselines are competitive with previous work, which includes D-Net Li et al. [2019] and Delphi Longpre et al. [2019], the top two systems of the MRQA shared task, while the 24-layer variants considerably outperform the current state of the art across all datasets. RoBERTa++ again performs better than RoBERTa on all datasets except DROP and RACE. DROP is designed to test arithmetic reasoning, while RACE contains (often fictional and thus not groundable to

	MRQA-In	BioASQ	TextbookQA	DuoRC	RE	DROP	RACE	MRQA-Out
Shared task								
D-Net (Ensemble)	84.82	-	-	-	-	-	-	70.42
Delphi	-	71.98	65.54	63.36	87.85	58.9	53.87	66.92
This work								
RoBERTa (Base)	82.98	68.80	58.32	62.56	86.87	**54.88**	**49.14**	68.17
RoBERTa++ (Base)	83.22	68.36	60.51	62.40	87.93	53.11	47.90	68.38
TEK$_F$ (Base)	83.44	69.71	62.19	63.43	87.49	51.04	46.43	68.46
TEK$_{PF}$ (Base)	**83.71**	**72.58**	**62.55**	**64.43**	**88.29**	54.58	47.75	**70.01**
RoBERTa (Large)	85.75	73.41	65.95	66.79	88.82	**68.63**	**56.84**	74.02
RoBERTa++ (Large)	85.80	74.73	67.51	67.40	**89.58**	67.62	55.95	74.58
TEK$_F$ (Large)	86.23	75.37	68.17	**68.80**	89.43	67.46	55.20	74.88
TEK$_{PF}$ (Large)	**86.33**	**76.80**	**69.10**	68.54	89.15	66.24	56.14	**75.00**

Table 4.3: In-domain and out-of-domain performance (F1) on MRQA. RE refers to the Relation Extraction dataset. MRQA-Out refers to the averaged out-of-domain F1.

Wikipedia) passages from English exams for middle and high school students in China. The performance drop after further pretraining on Wikipedia could be a result of multiple factors including the difference in style of required reasoning or content; we leave further investigation of this phenomenon for future work. The base variants of TEK$_F$ and TEK$_{PF}$ outperform both baselines on all other datasets. Comparing the base variant of our full TEK$_{PF}$ approach to RoBERTa++, we observe an overall improvement of 1.6 F1 with strong gains on BioASQ (4.2 F1), DuoRC (2.0 F1), and TextbookQA (2.0 F1). The 24-layer variants of TEK$_{PF}$ show similar trends with improvements of 2.1 F1 on BioASQ, 1.1 F1 on DuoRC, and 1.6 F1 on TextbookQA. Our large models see a reduction in the average gain mostly due to drop in performance on DROP. Like in the case of TriviaQA, TEK-pretraining generally improves performance even further where TEK-finetuning is useful (with the exception of DuoRC which sees a small loss of 0.24 F1 due to TEK-pretraining for the large models[3]), with the biggest gains seen on BioASQ.

Takeaways Both TEK$_{PF}$ and TEK$_F$ record strong gains on benchmarks that focus on factual reasoning outperforming the RoBERTa-based baselines that use only RC-contexts. The success of TEK$_F$ underscores the advantage of *textual* encyclopedic knowledge in that it improves current models even without additional TEK-pretraining. Finally, TEK-pretraining further improves the model's ability to use the retrieved

[3]According to the Wilcoxon signed rank test of statistical significance, the large TEK$_{PF}$ is significantly better than TEK$_F$ on BioASQ and TextbookQA p-value $< .05$, and is not significantly different from it for DuoRC.

		Pretraining	Finetuning	Wiki	Web	MRQA
t	1	Context-O	Context-O	72.8	81.2	83.2
	2	Context-O	TEK	74.2	82.4	83.4
	3	TEK	Context-O	72.9	81.6	83.3
	4	TEK	TEK	**75.1**	**82.8**	**83.7**

Table 4.4: Development set F1 on TriviaQA and MRQA for base models using different combinations of pretraining and finetuning. Metrics are average F1 over 3 random finetuning seeds.

	N_C	N_B	Wiki	Web	MRQA
t	384	0	72.4	80.4	83.0
	512	0	72.8	81.2	83.2
	384	128	**74.2**	**82.4**	**83.4**
	256	256	73.6	82.2	83.3
	128	384	68.1	79.5	81.7

Table 4.5: F1 on TriviaQA and MRQA dev sets for varying lengths of context (N_C) and background (N_B).

background knowledge for the downstream RC task.

4.4 Ablation Studies

Comparing TEK Pretraining and Context-only Pretraining We also compare the two pretraining setups for models which do *not* use background knowledge to form representations for the finetuning tasks. Table 4.4 shows results for all four combinations of the pretraining and finetuning method variables, using 12-layer base models on the development sets of TriviaQA and MRQA (in-domain). Comparing rows 1 and 3, we see marginal gains across all datasets for TEK pretraining indicating that pretraining with encyclopedic knowledge does not hurt QA performance even when such information is not available during finetuning and inference. While previous work Liu et al. [2019c]; Joshi et al. [2020a] has shown that pretraining with single contiguous chunks of text clearly outperforms BERT's bi-sequence pipeline,[4] our results suggest that using *background* sentences from other documents during pretraining has no adverse effect on the downstream tasks we consider.

Trade-off between Document Context and Knowledge Our approach incorporates textual knowledge by using a part of the Transformer window for it, instead of additional context from the same document.

[4]BERT randomly samples the second sequence from a different document in the corpus with a probability of 0.5.

	Wiki	Web	In	Out
RoBERTa++	71.7	81.4	83.2	68.4
t TEK$_{PF}$	76.0	83.4	83.7	70.0
TEK$_{PF}$-GC	75.4	83.0	83.6	69.4
TEK$_{PF}$-TagMe	75.6	83.1	83.7	69.7

Table 4.6: Performance (F1) of 12-layer TEK$_{PF}$ when used with publicly available entity linkers on TriviaQA test sets and MRQA in (In) and out-of-domain (Out).

Having established the usefulness of the background knowledge even without tailored pretraining, we now consider the trade-off between neighboring context and retrieved knowledge (Table 4.5). We first compare using a shorter window of 384 tokens for RC-contexts with using 512 tokens for RC-contexts (the first two rows). Using longer document context results in consistent gains, some of which our TEK-enriched representations need to sacrifice. We then consider the trade-off for varying values of context length N_C and background length N_B (rows 2-5). The partitioning of 384 tokens for context and 128 for background outperforms other configurations. Moreover, adding up to 256 tokens of background knowledge improves performance over using only document context. *This suggests that relevant encyclopedic knowledge from outside of the current document is more useful than long-distance neighboring text from the same document for these benchmarks.*

Choice of the Entity Linker Table 4.6 compares the performance of TEK$_{PF}$ when used with publicly available entity linkers, Google Cloud Natural Language API (abbreviated as GC)[5] and TagMe Ferragina and Scaiella [2010]. Using TagMe results in a minor drop of around 0.3 F1 from TEK$_{PF}$ across benchmarks while still maintaining major gains over RoBERTa++. The results indicate that the choice of entity linker can make a difference but our method is robust and performs well with multiple linkers.

4.5 Discussion

When are TEK-enriched representations most useful for question answering? Across all evaluation benchmarks, the strongest gains are on TriviaQA, BioASQ, and TextbookQA. All three datasets involve questions targeting the long tail of factual information, which has sizable coverage in Wikipedia, the encyclopedic

[5]https://cloud.google.com/natural-language/docs/basics#entity analysis

collection we use. We hypothesize that enriching representations with encyclopedic knowledge could be particularly useful when factual information that might be difficult to "memorize" during pretraining is important. Current pretraining methods are able to store a significant amount of world knowledge into model parameters Petroni et al. [2019]; this might enable the model to make correct predictions even from contexts with complex phrasing or partial information. TEK-enriched representations complement this strength

---

Question: Which river originates in the *Taurus Mountains*, and flows through *Syria and Iraq?*
Our Answer: Euphrates
Baseline Answer: Tigris
Context: The Southeastern *Taurus mountains* form the northern boundary... They are also the source of the Euphrates River and Tigris River.
Background: Originating in eastern Turkey, the Euphrates flows through *Syria and Iraq* to join the Tigris...

Question: What *tyrosine kinase*, involved in a Philadelphia- chromosome positive *chronic myelogenous leukemia*, is the target of *Imatinib (Gleevec)*?
Our Answer: BCR-ABL
Baseline Answer: imatinib
Context: Imatinib induces a durable response in most patients with Philadelphia chromosome-positive *chronic myeloid leukemia*...We show that the only hypothesis consistent with current data on ... gradual decrease in the BCR-ABL levels seen in most patients is that these patients exhibit a continual, gradual reduction of the LSCs. This observation may explain the ability to discontinue imatinib therapy without relapse in some cases.
Background: *Chronic myelogenous leukemia* : A 2006 follow up of 553 patients using imatinib (Gleevec) found an overall survival rate of 89% after five years. With improved understanding of the nature of the BCR-ABL protein and its action as a *tyrosine kinase*, targeted therapies (the first of which was imatinib) that specifically inhibit the activity of the BCR-ABL protein have been developed.

Question: Who did *Germany* defeat to win the *1990 FIFA World Cup*?
Our Answer: Argentina
Baseline Answer: Italy
Context: At the *1990 World Cup* in Italy, West *Germany* won their third World Cup title, defeating Yugoslavia (4-1), UAE on the way to a final rematch against Argentina.
Background: At international level, He is best known for scoring the winning goal for *Germany* in the *1990 FIFA World Cup* Final against Argentina...

Question: The *state* in which *matter* takes on the *shape* but not the *volume* of its *container* is?
Our Answer: Liquid
Baseline Answer: gas
Context: Liquid takes the *shape* of its *container*. You could put the same volume of liquid in containers with different shapes. The shape of the liquid in the beaker is short and wide like the beaker, while the shape of the liquid in the graduated cylinder is tall and narrow like that container, but each container holds the same *volume* of liquid... How could you show that gas spreads out to take the volume as well as the shape of its container?
Background: Liquid : As such, it is one of the four fundamental *states of matter* is the only state with a definite *volume* but no fixed *shape*.

Figure 4.3: The first two examples (from TriviaQA and BioASQ) have background knowledge that provides information complementary to the context, while the last two (from TriviaQA and TextbookQA) provides a more direct, yet redundant, phrasing of the information need compared to the original context.

via dynamic retrieval of factual knowledge. Finally, improvements on the science-based datasets BioASQ and TextbookQA further suggest that Wikipedia can be used as a *bridge corpus* for more effective domain adaptation for QA.

For 75% of the examples in the TriviaQA Wikipedia development set where our approach outperforms the context-only baselines, the answer string is mentioned in the background text. A qualitative analysis of these examples indicates that the retrieved background information typically falls into two categories – (a) where the background helps disambiguate between multiple answer candidates by providing partial pieces of information missing from the original context, and (b) where the background sentences help by providing a redundant but more direct phrasing of the information need compared to the original context. Figure 4.3 provides examples of each category.

Even when the retrieved background contains the answer string, our model uses the background only to refine representations of the candidate answers in the original document context; possible answer positions in the background are not considered in our model formulation. This highlights the strength of an encoder with full cross-attention between RC-contexts and background knowledge. The encoder is able to build representations for, and consider possible answers in all document passages, while integrating knowledge from multiple pieces of external textual evidence.

The exact form of background knowledge is dependent on the retrieval function. Our results have shown that contextualizing the input with textual background knowledge, especially after suitable pretraining, improves state of the art methods even with simple entity linking and *ngram*-match retrieval functions. We hypothesize that more sophisticated retrieval methods could further significantly improve performance (for example, by prioritizing for more complementary information).

4.6 Related Work

Background Knowledge Integration Many NLP tasks require the use of multiple kinds of background knowledge Fillmore [1976]; Minsky [1986]. Earlier work Ratinov and Roth [2009]; Nakashole and Mitchell [2015] combined features over the given task data with hand-engineered features over knowledge repositories. Other forms of external knowledge include relational knowledge between word or entity pairs, typically integrated via embeddings from structured knowledge graphs (KGs) Yang and Mitchell [2017]; Bauer et al.

[2018]; Mihaylov and Frank [2018]; Wang and Jiang [2019] or via word pair embeddings trained from text Joshi et al. [2019a]. Weissenborn et al. [2017] used a specialized architecture to integrate background knowledge from ConceptNet and Wikipedia entity descriptions. For open-domain QA, recent works Sun et al. [2019a]; Xiong et al. [2019] jointly reasoned over text and KGs, via specialized graph-based architectures for defining the flow of information between them. These methods did not take advantage of large scale unlabeled text to pre-train deep contextualized representations which have the capacity to encode even more knowledge in their parameters.

Most relevant to ours is work building upon these powerful pretrained representations, and further integrating external knowledge. Recent work focuses on refining pretrained contextualized representations using entity or triple embeddings from structured KGs Peters et al. [2019]; Yang et al. [2019a]; Zhang et al. [2019b]. The KG embeddings are trained separately (often to predict links in the KG), and knowledge from KG is fused with deep Transformer representations via special-purpose architectures. Some of these prior works also pre-train the knowledge fusion layers from unlabeled text through self-supervised objectives Zhang et al. [2019b]; Peters et al. [2019]. Instead of separately encoding structured KBs, and then attending to their single-vector embeddings, we explore directly using wider-coverage textual encyclopedic background knowledge. This enables direct application of a pretrained deep Transformer (RoBERTa) for jointly contextualizing input text and background knowledge. We showed background knowledge integration can be further improved by additional knowledge-augmented self-supervised pretraining.

Liu et al. [2019a] augment text with relevant triples from a structured KB. They process triples as word sequences using BERT with a special-purpose attention masking strategy. This allows the model to partially re-use BERT for encoding and integrating the structured knowledge. Our work uses wider-coverage textual sources instead and shows the power of additional knowledge-tailored self-supervised pretraining.

Question Answering For open-domain QA, where documents known to answer the question are not given as input (e.g. OpenBookQA Mihaylov et al. [2018]), methods exploring retrieval of relevant textual knowledge are a necessity. Recent work in these areas has focused on improving the evidence retrieval components Lee et al. [2019]; Banerjee et al. [2019]; Guu et al. [2020], and has used Wikidata triples with textual descriptions of Wikipedia entities as a source of evidence Min et al. [2019]. Other approaches use pseudo-relevance feedback (PRF) Xu and Croft [1996] style multi-step retrieval of passages by query reformulation Buck et al.

[2018]; Nogueira and Cho [2017], entity linking Das et al. [2019b], and more complex reader-retriever interaction Das et al. [2019a]. When multiple candidate contexts are retrieved for open-domain QA, they are sometimes jointly contextualized using a specialized architecture Min et al. [2019]. We are the first to explore pretraining of representations which can integrate background from multiple documents, and hypothesize that these representations could be further improved by more sophisticated retrieval approaches.

4.7 Conclusion

We presented a method to build text representations by jointly contextualizing the input with dynamically retrieved textual encyclopedic knowledge. We showed consistent improvements, in- and out-of-domain, across multiple reading comprehension benchmarks that require factual reasoning and knowledge well represented in the background collection.

Chapter 5

Few-shot Mining of Naturally Occurring Inputs and Outputs

Gathering high-quality training data has been one of the most reliable ways of achieving empirical progress for a range of natural language processing tasks. However, creating *natural* language training data often involves significant human effort – in carefully designing the data collection tasks as well as getting contributors to perform them. The complexity of this process in turn drives up the cost of creating training data. In this paper, we mine naturally-occurring examples from large corpora using supervision from a small seed set of only 100 labeled examples.

Our method provides a way to collect more training data using very few labeled examples, and allows for more direct control over what the model learns compared to relatively brittle prompts Lu et al. [2021]. Yet, unlike model generated data augmentation, we mine high-quality human-authored data which is less susceptible to the limitations of synthetic data. While similar unsupervised methods have been used for mining parallel data for machine translation Tran et al. [2020], we show that they can be extended to other tasks using minimal human supervision.

One of the main challenges in mining high-quality data is the large search space which could be quadratic in the size of the corpora as inputs and outputs could be spread across multiple documents. Our approach consists of a two-stage pipeline which first mines data efficiently from large corpora and then filters out low-quality examples. More specifically, we use supervision from an initial seed set and train a dot-product

Summary: The Scottish city of Edinburgh is looking to crack down on so-called "silent disco" walking tours as residents complain they make too much noise.
Passage: Silent disco tours in Edinburgh are 'too loud' and could face clampdown Silent discos in Edinburgh could be facing a clampdown - with locals upset that the crowds of boogying tourists are too loud. The "discos" involve people wearing their own headphones and dancing along as they follow a guided walking tour of the Scottish capital's most famous spots. ...

Question: How long does it take for a crab to get full grown?
Passage: 10-13 moults the crab will reach maturity. This usually takes *3-4 years*, but when food is limited it can take longer to reach maturity.

Figure 5.1: Examples of mined input output pairs for summarization and reading comprehension. Answer spans are indicated via italics.

similarity function over separately encoded fixed length dense representations of inputs and outputs (i.e. a dual-encoder or a biencoder). This function efficiently searches top-k candidate outputs (e.g. summary sentences) for each candidate input (e.g. news document) from the output corpus. We refer to this stage as coarse-grained search. It relies on maximal inner product search (MIPS) over dense encodings on inputs and outputs for efficiency. The initial list of paired inputs and outputs from this stage is geared towards recall and often contains examples with subtle errors. For example, it might link a document about World Cup 2006 with a summary sentence about World Cup 2010.

We further train a precision-oriented crossencoder-based filter which re-ranks the output of the first stage for better quality control. The crossencoder is trained using positive samples from the original seed set and negative examples from the coarse-grained stage. Jointly encoding inputs and outputs provides fine grained interaction between them. The crossencoder is able to filter out more subtle errors such as the one described earlier to give a high quality mined dataset set.

We apply our method to reading comprehension and summarization gathering high quality question-answer-passage and document-summary tuples respectively. The supervision from the small seed set helps embed relevant outputs closer to inputs across corpora for each task. This is in contrast to MT where cross-lingual cosine similarity functions are readily available without additional human supervision since self-supervised pre-training embeds sentences across languages in the same space.

Our method is able to add up to 5x as many examples as the seed set. On SQuAD Rajpurkar et al. [2016], augmenting the seed set with the mined data results in an improvement of 13 F1 over a BART-large Lewis et al. [2020b] baseline fine-tuned only on the seed set. Likewise, we see improvements of 1.46 ROUGE-L on XSum abstractive summarization Narayan et al. [2018]. Our analysis shows that, compared

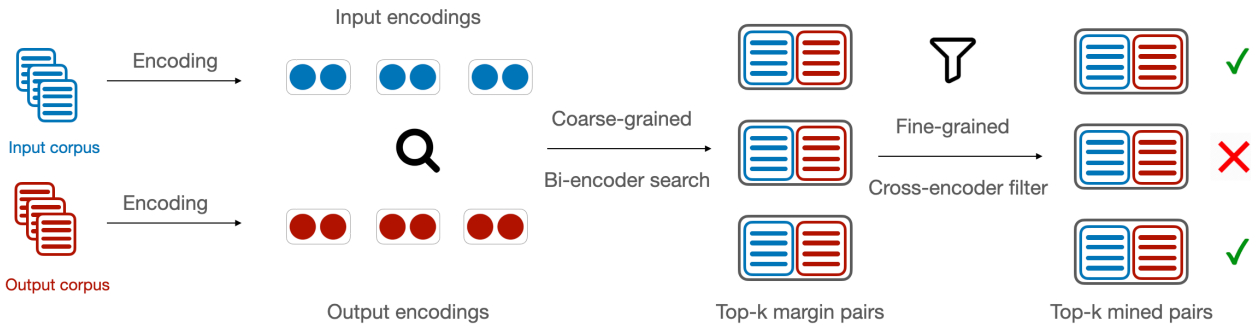

Figure 5.2: A pipeline of coarse and fine-grained models for mining high-quality data from large corpora. The input and output corpora are encoded separately using a biencoder. The coarse-grained search takes these encodings as input and produces input-output pairs which are then re-ranked by a crossencoder to produce high-quality data.

to model generated data, mined data better matches several characteristics of the gold data. For example, mined document-summary pairs are more abstractive and are topically closer to XSum.

5.1 Mining

We describe our general pipeline for mining examples, and then describe how it is applied to reading comprehension and summarization. We assume access to a small labeled seed set of 100 examples which can be used to train components of the pipeline. The final output is a (ranked) list of input output pairs which can be augmented to the original seed set to train better-performing models.

The key idea consists of a two stage approach for mining input output pairs (x, y) from their respective corpora C_x and C_y – (a) a coarse-grained but efficient similarity function which retrieves top-k similar candidates from C_y for every $x \in C_x$ and (b) a fine-grained filtering stage which reranks top scoring candidate pairs mined in the first stage.

5.1.1 Coarse-grained Search

The coarse grained search relies on dense retrieval to construct a candidate list of input-output pairs. We first train a function $f_c(\cdot, \cdot)$ to maximize similarity between input output pairs in the seed data, and then use the similarity function to mine the candidate input output pairs.

Training Coarse-grained Similarity

The aim is to create a vector space where the similarity between relevant input output pairs is higher than that between irrelevant pairs. Specifically, we use cosine similarity as our similarity function to enable efficient mining (Section 5.1.1).

Objective Let $D = \{(x_i, y_i^+, y_{i,1}^- ... y_{i,n}^-)\}_{i=1}^{i=m}$ be the training data that consists of m seed instances. Each instance contains one input x_i and one output y_i^+ along with n sampled negative outputs $y_{i,*}^-$. Our biencoder first encodes inputs and outputs separately as $\mathbf{x}_*$ and $\mathbf{y}_*$. The training objective maximizes the negative log likelihood of the positive output.

$$L(x_i, y_i^+, y_{i,1}^- ... y_{i,n}^-) = \frac{e^{\mathbf{x}_i \cdot \mathbf{y}_i^+}}{e^{\mathbf{x}_i \cdot \mathbf{y}_i^+} + \sum_{j=1}^{n} e^{\mathbf{x}_i \cdot \mathbf{y}_{i,j}^-}}$$

We use in-batch negatives along with randomly sampled sentences from the output corpus to construct $y_{i,*}^-$.

Mining Candidate Pairs

To mine input output pairs, we follow previous work on unsupervised machine translation and employ the margin function formulation Artetxe and Schwenk [2019]; Tran et al. [2020] based on K nearest neighbors (KNN). Let $\mathbf{x}$ and $\mathbf{y}$ be the vector representations of a candidate input output pair (x, y). We score the "similarity" of x and y using a ratio margin function defined as the following.

$$score(x, y) = \frac{cos(\mathbf{x}, \mathbf{y})}{\sum_{z \in N_x} \frac{cos(\mathbf{x}, \mathbf{z})}{2k} + \sum_{z \in N_y} \frac{cos(\mathbf{z}, \mathbf{y})}{2k}}$$

where N_x is the KNN neighborhood of x in C_y, the corpus of y; and N_y is the KNN neighborhood of y in C_x. The margin scoring function is interpreted as a cosine score normalized by average distances to the margin regions established by the KNN neighborhoods of the input and outputs. The KNN distance metrics are defined by $cos(x, y)$. The margin score is designed to take into account scale inconsistencies in cosine similarity, and has been shown to perform better than approaches which use a hard threshold over cosine similarity. We use the distributed dense vector similarity search library FAISS Johnson et al. [2019]

to search for all neighborhoods efficiently.

5.1.2 Fine-grained Filtering

We found the quality of the biencoder-mined data to be rather low for summarization and reading comprehension. Intuitively, separate encoding of inputs and outputs combined with cosine based scoring makes it much harder to control characteristics like abstraction for summarization. The fine grained filtering stage re-ranks the outputs of the previous stage using a crossencoder based ranking function to remove noisy input-output pairs. A crossencoder jointly encodes the input and output text enabling more fine grained interaction between them. Specifically, we encode a pair (x, y) as sequence $[CLS]$ x $[SEP]$ y $[SEP]$ where $[CLS]$ and $[SEP]$ are special tokens indicating beginning and end of sequences. The same input is fed into the encoder and decoder. The pair is scored using a 2-layer MLP on top of the final hidden state of the final decoder token. A more detailed comparison of the data obtained from the two stages is presented in Section 5.5.

5.2 Application to Tasks

One of the main challenges in mining data is the scale of web corpora. In this section, we describe additional techniques which make our method scalable when applied to downstream tasks.

5.2.1 Reading Comprehension

Inputs and Outputs For reading comprehension, the input sequence X consists of the question and the output sequence Y consists of a concatenation of the answer and the passage. Our input corpus consists of 20M random questions crawled from the popular community QA website *answers.com*. The output corpus consists of passages from Wikipedia with named entities[1] as candidate answers resulting in 200M passage-answer pairs. Our input and output corpora were chosen to match the downstream evaluation data from SQuAD.

[1] We use spaCy NER.

Binary Classifier To mitigate noise in the question corpus and to reduce the search space, we multitask a binary classifier with the biencoder to filter out questions which don't look like those in SQuAD. The binary classifier is trained using questions from the seed set as positive examples with negatives sampled from the question corpus.

Furthermore, for faster processing, we filter out examples from the coarse grained stage, where the answer spans are found verbatim in the question. Finally, while fine-tuning using mined (and seed) data, we add a special token to the input document indicating the provenance of the example.

5.2.2 Summarization

Inputs and Outputs For summarization, the input sequence X consists of the document and the output sequence Y consists of the summary. The input corpus consists of documents from CC-News; we remove documents with less than four sentences to reduce noise. The output corpus consists of sentences from the same corpus. Following MARGE Lewis et al. [2020a], we divide CC-News into shards based on document publishing dates, and restrict the space of candidate summaries for a given document to be from the same shard.

Binary Classifier Like reading comprehension, we multitask a binary classifier with the biencoder to remove sentences which don't resemble XSum summaries. This allows us to filter out more than 80% of the sentences and makes the biencoder based mining a lot more scalable.

Finally, for faster processing, we filter out examples from the coarse grained stage, where the summary sentences are found verbatim in the document. Put together, this allows use to efficiently search over 40M documents and 550M sentences.

5.3 Experiments

5.3.1 Benchmarks

We perform experiments on SQuAD v1.1 for reading comprehension and XSum for summarization – two competitive benchmarks which are used to evaluate state of the art pre-trained models. For each dataset, we randomly pick 100 examples from the training set as our seed set; we call this subset X_{100}.

Method	EM	F1
BART	35.27	49.43
Us	49.39	62.5

Table 5.1: Performance on the SQuAD dev set after training on 100 examples.

Method	R-1	R-2	R-L
BART	36.28	14.07	27.98
Us	37.67	14.86	29.44
BART*	35.17	13.29	27.20
WikiTransfer*	37.26	14.20	28.85
Pegasus*	39.07	16.44	31.27

Table 5.2: Performance on the XSum test set after training on 100 examples. Both BART and Us use the same training examples. * indicates that results have been taken from Fabbri et al. [2021]

SQuAD SQuAD Rajpurkar et al. [2016] is an extractive reading comprehension benchmark containing Wikipedia passages and crowdsourced questions.

Xsum Xsum Narayan et al. [2018] is an abstractive news summarization dataset collected by harvesting online articles from the British Broadcasting Corporation (BBC).

5.3.2 Baselines

BART Our main baseline for each dataset is a BART model fine-tuned on X_{100} and is referred to as BART in the subsequent tables.

For summarization, we also compare our method to WikiTransfer Fabbri et al. [2021] and Pegasus Zhang et al. [2020] – both of which use data augmentation techniques tailored towards summarization.

WikiTransfer WikiTransfer uses an intermediate finetuning stage with 400K synthetic examples from Wikipedia along with additional auxiliary losses before finetuning on Xsum.

Pegasus Pegasus, a self-supervised pretraining method, simulates the summarization tasks during pre-training by masking (and then generating) important heuristically identified sentences from documents.

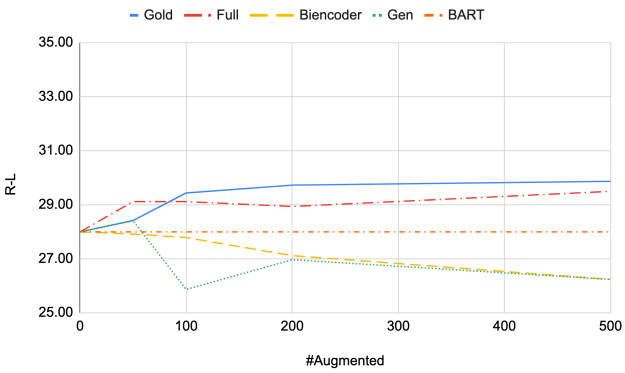

Figure 5.3: Performance (ROUGE-L) on the XSum dev set of various augmentation techniques with varying amounts of augmented data added to X_{100}.

5.3.3 Main Results

SQuAD Table 5.1 shows our performance on the SQuAD v1.1 dev set. The BART baseline reaches an F1 of 48.86; adding 500 mined examples gives us a boost of 13 F1 points.

XSum Table 5.2 shows our performance on the Xsum test set. Augmenting the 100 example training set with 500 mined examples increases the ROUGE-L score of the fine-tuned BART baseline from 27.98 (BART) to 29.44 (Us).

BART* refers to BART results taken from WikiTransfer.[2] Our gains over the BART baseline are comparable to WikiTransfer despite using only 500 augmented examples indicating that our examples are high-quality. It is harder to make more direct controlled comparisons to Pegasus which uses a larger model and a tailored pre-training scheme compared to BART.

Our approach is complementary to both WikiTransfer and Pegasus; in contrast to pre-training (or intermediate fine-tuning) on synthetic or heuristic-collected examples, we focus on mining fewer high quality examples aimed at mirroring the seed set so that they can be used directly during finetuning instead of pre-training.

[2]We posit that our numbers for BART are slightly higher due to a different seed set.

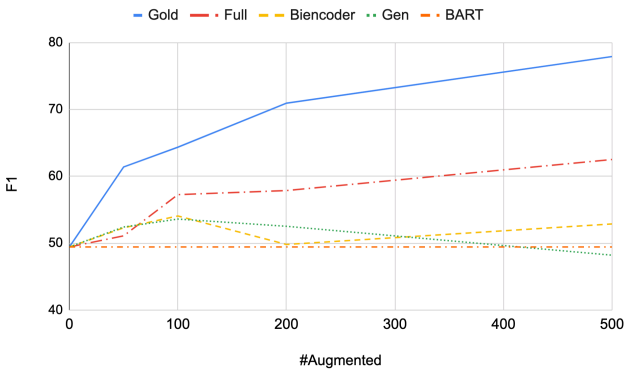

Figure 5.4: Performance (F1) on the SQuAD dev set of various augmentation techniques with varying amounts of augmented data added to X_{100}.

5.4 Ablations

In this section, we investigate the impact of each part of our pipeline and compare it to model-generated data augmentation (Figures 5.3 and 5.4). All methods involve fine-tuning BART on the same seed set plus the exact same number of augmented examples, but differ on the source of the augmented data.

Gold The Gold method augments gold examples sampled uniformly at random from the full training set of each dataset.

Full Full refers to examples obtained by running our full pipeline including the fine-grained crossencoder stage.

Biencoder This refers to post-processed examples from the coarse-grained biencoder which are subsequently fed to the crossencoder (in our full model).

Gen Gen refers to self-training on model generated examples. Specifically, we take source documents (or answer passage pairs in case of reading comprehension) from the full/crossencoder stage and generate summaries (or questions) using the BART baseline trained on X_{100}. We then retrain the model by augmenting X_{100} with the generated examples. This allows us to make a controlled comparison to the full model which uses naturally occurring mined examples.

5.4.1 Amount of Mined Data

For both SQuAD and XSum, we see an increase in performance for the Full model as we increase the number of mined examples up to 5x. There is still a considerable difference between our method and Gold indicating that there is still room for improvement in quality. We discuss these qualitative differences in Section 5.5.

5.4.2 Bi vs Crossencoder Differences

For SQuAD, input output pairs from the biencoder provide some improvement with 50 examples but performance quickly plateaus with more examples. For XSum, adding examples from the biencoder hurts performance. On the other hand, performance continues to improve until 5x examples from the crossencoder are added for both tasks. Section 5.5 contains a qualitative analysis of the bi and crossencoder examples.

5.4.3 Naturally Occurring vs. Model Generated Data

To show the value of mining naturally occurring data, we compare our method to the *Gen* baseline described in Section 5.4. For both XSum and SQuAD, adding model-generated examples provides some gains over BART, but performance quickly degrades with more examples. While model generated data augmentation has been shown to help for QA Alberti et al. [2019], our results suggest that it isn't particularly useful when training data is limited.

5.5 Discussion

We conducted a case study of the top ranking examples from the bi and cross encoders for both SQuAD and XSum (Table 5.3).

5.5.1 XSum

After examining outputs from the bi and crossencoder, we found that both stages were generally able to match documents to sentences from the same news story, but with the sentences taken from a parallel document. Exceptions to this included cases where similar events (e.g. World Cup games) occuring over different

Biencoder	Crossencoder
Summary: BERLIN (AP) — A 44-pound minute hand has fallen off a clock on a Hamburg church tower, plunging 130 feet onto the sidewalk below. **Document**: Minute hand plunges from Hamburg church tower; no one hurt BERLIN (AP) — A 20-kilogram (44-pound) minute hand has fallen off a clock on a Hamburg church tower, plunging 40 meters (130 feet) onto the sidewalk below...	**Summary**: Blackpool's three historic piers have been put on an international preservation list amid fears their future could be under threat from climate change. **Document**: ...The council is now completing an action plan to work with the World Monuments Fund which was launched in 1996 with founding partner American Express and issues a list every two years. Blackpool is unique in being the only UK seaside resort with three piers...
Summary: NEW LONDON, Conn. (AP) — The U.S. Coast Guard Academy is now offering an academic program in cyber systems, its first new major in a quarter century. **Document**: Coast Guard Academy to offer new major in cyber systems NEW LONDON, Conn. (AP) — The U.S. Coast Guard Academy is offering a new major in cyber systems. It's the first new academic program at the school in New London since the addition of mechanical engineering as a major in 1993...	**Summary**: Holidaymakers are being reminded to drink responsibly at Gatwick Airport as an annual campaign to tackle disruptive passengers is launched. **Document**: The force, working closely with the airport, its pubs and bars, and airlines, will carry out increased patrols as part of Operation Disrupt..."You could be refused carriage or sent on the next plane home if you are considered to be drunk, disorderly or disruptive...
Question: What is most times one actor has been nominated? **Answer/Passage**: The actress has also been nominated *five* times (most	**Question**: When was Regina Lobiondo born? **Answer/Passage**: Other notable appearances include a recurring role in L.A. Law, a regular role in the 1993 The Untouchables television series, and starring in the 1996 film It's My Party. Regina was born on *October 25, 1956*, in Brooklyn, New York.
Question: Where was Jesus baptised? **Answer/Passage**: He was baptised at *St Jude's*	**Question**: When was Gouverneur Morris born? **Answer/Passage**: ...Gouverneur Morris Jr. Gouverneur Morris II (*February 9, 1813* ž013 August 20, 1888) was an American railroad executive...

Table 5.3: Mined examples from the bi and crossencoders. Answer spans are indicated in *italics*.

time frames. However, we found three clear differences which could explain why data from the crossencoder is more helpful. First, summaries from the crossencoder were more abstractive than the biencoder. We measured abstractiveness using ROUGE precision scores with respect to the *source*; higher scores being indicative of higher extractiveness and lower abstractiveness. ROUGE-1/2/L precision scores for the crossencoder summaries (64.91 / 15.42 / 55.59) were closer to the XSum validation (66.85 / 17.94 / 59.82) compared to those from the biencdoer (82.21 / 51.66 / 75.92). Second, the distribution of topics and locations from the top ranked crossencoder documents better matched XSum with most stories centered around UK politics and sports. On the other hand, top ranked pairs from the biencoder were more diverse topically. Lastly, the summaries mined by the biencoder rarely matched the style of XSum and often contained source markers (e.g. CNN or AP News). Our analysis suggests that the biencoder uses word overlap as it's primary signal for scoring document summary pairs while the crossencoder is able to focus on finer details.

5.5.2 SQuAD

We found mining high-precision QA pairs to be a much harder problem in part due to the large search space of over 200M (answer, passage) tuples. Like summarization, biencoder outputs for QA focused on word overlap which resulted in the retrieval of very short passages (see Table 5.3). While the crossencoder was more successful in selecting longer passages, we still found pairs with subtle errors. For example, for the question, "When was Regina Lobiondo born?", the retrieved passage actually refers to Paul Regina and not Regina Lobiondo. We hypothesize the negative impact of such noise is mitigated in the reading comprehension setting where the given evaluation passage always answer the question. Another limitation of data from both stages was the lack of diversity in question types with a significant proportion of questions seeking date or numerical answers.

5.6 Related Work

Data augmentation Data augmentation is a popular technique with a large body of work Wang and Yang [2015]; Jia and Liang [2016] among others. Recent work has explored model generated data augmentation for a range of tasks including text classification Anaby-Tavor et al. [2020]; Schick and Schütze [2021a], question answering Alberti et al. [2019], common-sense reasoning Yang et al. [2020], and machine transla-

tion Sennrich et al. [2016]. A common problem with model-generated data augmentation is the quality of the synthetic data. Attempts to remedy this have focused on variations of consistency Xie et al. [2019] for a given task–such as round-trip consistency of question generation and answer prediction Alberti et al. [2019]; Puri et al. [2020] for QA or between source and targets in summarization Fabbri et al. [2021]. Lee et al. [2021] focus on generating synthetic data for underrepresented or few-shot slices. Task augmentation Vu et al. [2021] generates data in the target domain by using a model trained on the auxiliary task of natural language inference. In contrast, we focus on a general framework for mining *naturally occurring* data for multiple tasks using supervision from a small labeled seed set.

Improving fine-tuning Work in this area has looked at improving finetuning particularly for small datasets either via better optimization or by using auxiliary tasks. Careful design and hyperparameter choices have a significant impact on performance and stability in limited data settings Mosbach et al. [2021]; Zhang et al. [2019a]. Likewise, regularization based approaches have also been shown to improve performance Jiang et al. [2020]; Aghajanyan et al. [2020]. While we use lessons about careful design choices, our basic models use standard fine-tuning for simplicity. Efforts on the data side have focused on intermediate fine-tuning either by using unlabeled target domain data Karouzos et al. [2021]; Gururangan et al. [2020] or via labeled data from other tasks Phang et al. [2018]; Aghajanyan et al. [2021]; Vu et al. [2020].

Few-shot learning GPT-3 Brown et al. [2020] is perhaps the most prominent recent work in this area which shows that massive language models can perform a variety of tasks if prompted with few input output pairs. Other work has shown that good few-shot performance can also be obtained by combining gradient based optimization with textual prompts Schick and Schütze [2021b]; Gao et al. [2021]; Tam et al. [2021]; Wang et al. [2021]. Other work on pre-training focuses on tailored masking schemes such as recurrent span masking Ram et al. [2021] for question answering and salient sentence masking for summarization Zhang et al. [2020]. Our approach is complementary to this line of work in that it is agnostic to pre-training schemes or textual prompts, and focuses instead on supervised automated collection of high-quality data.

5.7 Conclusion

We presented a method to mine input output pairs from large corpora from supervision from a small seed set of labeled examples. We showed consistent improvements on XSum and SQuAD, two popular NLP benchmarks, from adding up to 5x examples to the seed set. Our analysis shows that, compared to model generated data, mined data better matches several characteristics of the gold data (e.g. abstractiveness in summarization).

Chapter 6

Conclusion

This thesis presents training paradigms for efficient self-supervised models centered around pre-training objectives, model scale, and new types of data. In the previous chapters, we presented work with the following goals: (a) designing efficient pre-training methods to capture linguistic and world knowledge, and (b) enabling better downstream performance with fewer human-labeled examples.

In Chapter 2, we discussed how designing the right kind of objective function can help encode knowledge about word pairs into word pair vectors. Our pair embeddings are computed as a compositional function of the individual word representations, which is learned by maximizing a variant of the PMI with the contexts in which the the two words co-occur. Bouraoui et al. [2020] have suggested that BERT based models do an even better job at capturing word pair relations but there is still room for improvement. Other methods Levine et al. [2021] have considered using PMI-based masking schemes for BERT for more efficient pre-training.

In Chapter 3, we present SpanBERT, a scalable pre-training method to encode encode more diverse kinds of knowledge into model parameters. Our approach extends BERT by (1) masking contiguous random spans, rather than random tokens, and (2) training the span boundary representations to predict the entire content of the masked span, without relying on the individual token representations within it. Subsequent work on self-supervised NLP models, T5 Raffel et al. [2020] and BART Lewis et al. [2020b] has used span-based masking as a part of their sequence-to-sequence pre-training. Other works have looked at improving span representations for coreference resolution Gandhi et al. [2021] and other tasks Toshniwal et al. [2020].

Chapter 4 focuses integrating dynamically retrieved textual knowledge. Our method represents input texts by contextualizing them jointly with dynamically retrieved textual encyclopedic background knowledge from multiple documents. We apply our method to reading comprehension tasks by encoding questions and passages together with background sentences about the entities they mention. Borgeaud et al. [2022] present a specialized architecture for integrating background about a given text without using entity-focused retrieval.

6.1 Future Work

On a more speculative note, we expect ideas from this thesis to drive work in the following directions.

Retrieval Augmented NLP Retrieval Augmented NLP has received considerable attention from the point of view of learning retrieval Guu et al. [2020] as well as integration of retrieved text Joshi et al. [2020b]. Yet most applications of retrieval focus on question answering and language modeling. Subsequent research could look at a more general framework for using retrieval for arbitrary NLP tasks. For example, news summarization is typically viewed as a task which does not need context beyond the provided document(s). Yet abstractive headlines often contain facts which aren't directly mentioned in the given documents. Using retrieval in the loop could help models generate summaries which require such additional world knowledge An et al. [2021]. We posit that a single retrieval in the loop framework for multiple NLP tasks could significantly improve the performance and adoption.

Multimodal Pre-training Recent work has looked at combining text and image data for pre-training a single model Aghajanyan et al. [2022]. An area for subsequent research could look into expanding this in terms of both scale and modalities. In particular, modalities like text and video could provide more direct evidence for several kinds of knowledge which are often more implicit in text, e.g. common sense.

Massively Multitasked Pre-training The advent of large scale pre-training has also resulted the use of a common encoder-decoder architectures for almost all NLP tasks. This, combined with common data mining techniques (Chapter 5), presents an opportunity for massively multitasked pre-training. The goal of this line

of research would be to move towards common representations with improved few-shot performance across multiple NLP tasks.

Bibliography

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5799–5811, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Armen Aghajanyan, Bernie Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, and Luke Zettlemoyer. 2022. Cm3: A causal masked multimodal model of the internet. *ArXiv*, abs/2201.07520.

Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. 2020. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*.

Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A neural knowledge language model. *arXiv preprint arXiv:1608.00318*.

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Chen An, Ming Zhong, Zhichao Geng, Jianqiang Yang, and Xipeng Qiu. 2021. Retrievalsum: A retrieval enhanced framework for abstractive summarization. *ArXiv*, abs/2109.07943.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Pratyay Banerjee, Kuntal Kumar Pal, Arindam Mitra, and Chitta Baral. 2019. Careful selection of knowledge to solve open book question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6120–6129, Florence, Italy. Association for Computational Linguistics.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, pages 6–4.

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230, Brussels, Belgium. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas,

Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7456–7463.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Andrea Gesmundo, Neil Houlsby, Wojciech Gajewski, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In *International Conference on Learning Representations*, Vancouver, Canada.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *International Workshop on Semantic Evaluation (SemEval)*, pages 1–14, Vancouver, Canada.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. KERMIT: Generative insertion-based modeling for sequences. *arXiv preprint arXiv:1906.01604*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 2406–2417. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668. Association for Computational Linguistics.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855. Association for Computational Linguistics.

Tim Van de Cruys. 2011. Two multivariate generalizations of pointwise mutual information. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 16–20. Association for Computational Linguistics.

Tim Van de Cruys. 2014. A neural network approach to selectional preference acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 26–35. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3079–3087. Curran Associates, Inc.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Association for Computational Linguistics (ACL)*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019a. Multi-step retriever-reader interaction for scalable open-domain question answering. In *International Conference on Learning Representations*.

Rajarshi Das, Ameya Godbole, Dilip Kavarthapu, Zhiyu Gong, Abhishek Singhal, Mo Yu, Xiaoxiao Guo, Tian Gao, Hamed Zamani, Manzil Zaheer, and Andrew McCallum. 2019b. Multi-step entity-centric information retrieval for multi-hop question answering. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 113–118, Hong Kong, China. Association for Computational Linguistics.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems (NIPS)*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Luis Espinosa Anke and Steven Schockaert. 2018. Seven: Augmenting word embeddings with unsupervised relation vectors. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2653–2665. Association for Computational Linguistics.

Alexander Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. 2021. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 704–717, Online. Association for Computational Linguistics.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *ACM International Conference on Information and Knowledge Management*, CIKM '10, page 1625–1628.

Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, 280(1):20–32.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*.

Nupoor Gandhi, Anjalie Field, and Yulia Tsvetkov. 2021. Improving span representation for domain-adapted coreference resolution. *ArXiv*, abs/2109.09811.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: What works and what doesn't. In *Proceedings of the NAACL-HLT SRW*, pages 47–54, San Diego, California, June 12-17, 2016. ACL.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Association for Computational Linguistics (ACL)*, pages 364–369.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339. Association for Computational Linguistics.

Shoaib Jameel, Zied Bouraoui, and Steven Schockaert. 2018. Unsupervised learning of distributional rela-

tion vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23–33. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. Association for Computational Linguistics.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020a. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Omer Levy, Daniel Weld, and Luke Zettlemoyer. 2019a. pair2vec: Compositional word-pair embeddings for cross-sentence inference. In *North American Association for Computational Linguistics (NAACL)*, pages 3597–3608.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics (ACL)*, pages 1601–1611.

Mandar Joshi, Kenton Lee, Yi Luan, and Kristina Toutanova. 2020b. Contextualized representations using textual encyclopedic knowledge. *ArXiv*, abs/2004.12006.

Mandar Joshi, Omer Levy, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019b. BERT for coreference resolution: Baselines and analysis. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. UDALM: Unsupervised domain adaptation through language modeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2579–2590, Online. Association for Computational Linguistics.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems (NIPS)*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics (TACL)*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Advances in Neural Information Processing Systems (NIPS)*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Kenton Lee, Kelvin Guu, Luheng He, Timothy Dozat, and Hyung Won Chung. 2021. Neural data augmentation via example extrapolation. *ArXiv*, abs/2102.01335.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 188–197.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *North American Association for Computational Linguistics (NAACL)*, pages 687–692.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. 2021. Pmi-masking: Principled masking of correlated spans. *ArXiv*, abs/2010.01825.

Omer Levy, Ido Dagan, and Jacob Goldberger. 2014. Focused entailment graphs for open ie propositions. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 87–97. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2177–2185, Cambridge, MA, USA. MIT Press.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), Vancouver, Canada, August 3-4, 2017*, pages 333–342.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. *ArXiv*, abs/2006.15020.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis. 2018. *Setting the TriviaQA SoTA with Contextualized Word Embeddings and Horovo*.

Bofang Li, Tao Liu, Zhe Zhao, Buzhou Tang, Aleksandr Drozd, Anna Rogers, and Xiaoyong Du. 2017. Investigating different syntactic context types and context representations for learning word embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2421–2431. Association for Computational Linguistics.

Hongyu Li, Xiyuan Zhang, Yibing Liu, Yiming Zhang, Quan Wang, Xiangyang Zhou, Jing Liu, Hua Wu, and Haifeng Wang. 2019. D-NET: A pre-training and fine-tuning framework for improving the generalization of machine reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 212–219, Hong Kong, China. Association for Computational Linguistics.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2019a. K-bert: Enabling language representation with knowledge graph. *arXiv preprint arXiv:1909.07606*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. RoBERTa: A robustly optimized BERT pretraining approach. *arxiv preprint arXiv:1907.11692*.

Robert Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual*

Meeting of the Association for Computational Linguistics, Florence, Italy. Association for Computational Linguistics.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. *arxiv preprint arXiv:1803.02893*.

Shayne Longpre, Yi Lu, Zhucheng Tu, and Chris DuBois. 2019. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 220–227, Hong Kong, China. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *ArXiv*, abs/2104.08786.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Computational Natural Language Learning (CoNLL)*, pages 51–61.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013a. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Marvin Minsky. 1986. *The Society of Mind*. Simon & Schuster, Inc., New York, NY, USA.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *ArXiv*, abs/2006.04884.

Ndapandula Nakashole and Tom M. Mitchell. 2015. A knowledge-intensive model for prepositional phrase attachment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 365–375, Beijing, China. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *North American Association for Computational Linguistics (NAACL)*, pages 48–53.

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

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *ArXiv*, abs/1811.01088.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163. Association for Computational Linguistics (ACL).

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3066–3079, Online. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 147–155, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, Benjamin M. Marlin, and Andrew McCallum. 2013. Relation extraction with matrix factorization and universal schemas. In *Joint Human Language Technology Conference/Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Enrico Santus, Anna Gladkova, Stefan Evert, and Alessandro Lenci. 2016. The CogALex-V shared task on the corpus-based identification of semantic relations. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*, pages 69–79. The COLING 2016 Organizing Committee.

Timo Schick and Hinrich Schütze. 2021a. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Vered Shwartz and Ido Dagan. 2016. Path-based vs. distributional information in recognizing lexical semantic relations. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon, CogALex@COLING 2016, Osaka, Japan, December 12, 2016*, pages 24–29.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398. Association for Computational Linguistics.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press.

Livio Baldini Soares, Nicholas Arthur FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Association for Computational Linguistics (ACL)*, pages 2895–2905.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning (ICML)*, pages 5926–5936.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems (NIPS)*.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019a. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390, Hong Kong, China. Association for Computational Linguistics.

Yu Stephanie Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xinlun Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019b. ERNIE: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4980–4991, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Shubham Toshniwal, Haoyue Shi, Bowen Shi, Lingyu Gao, Karen Livescu, and Kevin Gimpel. 2020. A cross-task analysis of text span representations. In *REPL4NLP*.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1499–1509.

C. Tran, Y. Tang, Xian Li, and Jiatao Gu. 2020. Cross-lingual retrieval for iterative self-supervised training. *ArXiv*, abs/2006.09526.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *2nd Workshop on Representation Learning for NLP*, pages 191–200.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16(1):138.

Peter D. Turney. 2005. Measuring semantic similarity by latent relational analysis. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 1136–1141, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*.

Tu Vu, Minh-Thang Luong, Quoc Le, Grady Simon, and Mohit Iyyer. 2021. STraTA: Self-training with task augmentation for better few-shot learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5715–5731, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Chao Wang and Hui Jiang. 2019. Explicit utilization of general knowledge in machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2263–2272, Florence, Italy. Association for Computational Linguistics.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. Entailment as few-shot learner. *ArXiv*, abs/2104.14690.

Wei Wang, Ming Yan, and Chen Wu. 2018. Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1705–1714, Melbourne, Australia. Association for Computational Linguistics.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, Lisbon, Portugal. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Koki Washio and Tsuneaki Kato. 2018a. Filling missing paths: Modeling co-occurrences of word pairs and dependency paths for recognizing lexical semantic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1123–1133. Association for Computational Linguistics.

Koki Washio and Tsuneaki Kato. 2018b. Neural latent relational analysis to capture lexical semantic relations in a vector space. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*. Association for Computational Linguistics.

Dirk Weissenborn, Tomáš Kočiskỳ, and Chris Dyer. 2017. Dynamic integration of background knowledge in neural nlu systems. *arXiv preprint arXiv:1706.02596*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018a. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. In *North American Association for Computational Linguistics (NAACL)*, pages 1112–1122.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete KBs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4258–4264, Florence, Italy. Association for Computational Linguistics.

Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In *SIGIR*.

An Yang, Quan Wang, Jing Liu, Kai Liu, Yajuan Lyu, Hua Wu, Qiaoqiao She, and Sujian Li. 2019a. Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in LSTMs for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1436–1446. Association for Computational Linguistics.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. Generative data augmentation for commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1008–1025, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019b. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *ArXiv*, abs/1912.08777.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav. Artzi. 2019a. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 35–45.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019b. ERNIE: Enhanced

language representation with informative entities. In *Association for Computational Linguistics (ACL)*, pages 1441–1451.