Erasmus University Rotterdam

Erasmus School of Economics

Master Thesis Business Analytics and Quantitative Marketing

DAWSON: Data Augmentation using Weak Supervision On Natural Language

---

Tim de Jonge van Ellemeet

425288

April 30, 2021

Supervisor: Dr. Flavius Frăsincar

Second assessor: Dr. Wendun Wang

Abstract

We propose a novel data augmentation model for text, using weak supervision. Recent work in the field uses BERT and masked language modeling to conditionally augment data. These models all involve a small, high-quality labeled dataset, but omit the abundance of unlabeled data, which is likely to be present if one considers a model in the first place. Weak supervision methods, such as Snorkel, make use of the vastness of unlabeled data, but largely omit the available ground truth labels. We combine data augmentation and weak supervision techniques into a holistic method, consisting of 4 training phases and 2 inference phases, to efficiently train an end-to-end model when only a small amount of annotated data is available. We outperform the benchmark (Kumar et al., 2020) for the SST-2 task by 1.5, QQP task by 4.4, and QNLI task by 3.0 absolute accuracy points, and - for the first time - show that data augmentation is also effective for natural language understanding tasks, such as QQP and QNLI.

Contents

List of Figures

List of Tables

1 Introduction

The advent of deep neural networks and transfer learning has fundamentally changed the challenges of Machine Learning (ML). In natural language processing (NLP), task-specific vocabulary construction, text cleaning, and model architectures have been rendered mostly obsolete by transformer models (Vaswani et al., 2017). However, as model architectures have grown larger, so did the amount of data required to train them. The limiting factor has become the collection of high-quality training data (Hancock et al., 2019). In this paper, we focus on the common situation, in which there is only a small dataset with high-quality labels, but an abundance of unlabeled data. We present novel techniques to extract more information out of all data available.

There are two primary methods for handling situations with little annotated data: data augmentation (DA) & weak supervision (WS). In data augmentation, high-quality labeled samples are augmented to create new samples, while entirely omitting the large unlabeled dataset. In weak supervision, external knowledge bases, related datasets, or rules of thumb are used to generate low-quality label estimates for a large collection of unlabeled data. High-quality labeled data - if available - is typically used for validation only. Both methods aim to solve the same problem, but are rarely found together in academic research. Therefore we pose the research question:

How can a large amount of weakly-supervised data and a small amount of high-quality labeled data be combined to improve model performance in NLP?

In this work, we propose to combine data augmentation, weak supervision and recent transformer research into a holistic methodology that - to the best of our knowledge - is a new contribution to the field. We present the methodology as Data Augmentation using Weak Supervision On Natural Language (DAWSON). Throughout this work, we use the following negative movie review as running example:

"one relentlessly depressing situation"

In Table 1.1, the input and output of the augmentation model are given for the running example. The end result of our methodology is a *dataset*, which is a combination of both the original and augmented texts. The aim of this work is to improve the augmentations by adding additional training steps to obtain a better augmentation model. In total, there are 4 training phases and 2 inference phases.

Table 1.1: The input and output of a conditional augmentation model.

Input (text):	*one relentlessly depressing situation*
Input (label):	**negative**
Output:	*one relentlessly **boring** situation*

In the following sections, we introduce the models and methods that are used in this work. First, an introduction to transformers and transfer learning is given. Next, the need for conditional data augmentation and the role of a transformer model are explained. Furthermore, the concepts of weak supervision and span extraction are introduced. Last, a summary of the proposed methodology and contribution is given.

1.1 BERT

With *Attention is All You Need*, Vaswani et al. (2017) introduced a new paradigm for NLP. At the heart of the transformer architecture lies the concept of self-attention. Attention can be seen as a probability distribution over all words in a sentence, representing how much each word influences the target word. Attention is needed, such that a model can incorporate *context*. Self-attention is a novel method for computing attention. Each word has a trainable *query*, *key*, and *value* vector. The attention is calculated as the dot products between the query vector of the target word, and the key vectors of all words in the sentence. After applying softmax, the resulting distribution is used as weights for the weighted sum of all value vectors, of which the output is the new representation of the target word. In a transformer block, self-attention is computed multiple times simultaneously (multi-headed attention), and pooled using dense neural network layers. Furthermore, in a block, neural networks are used to further process word vectors, resulting in a final numerical representation, also known as an *embedding*. For a more detailed explanation of transformers, we refer to the excellent blog by Alammar (2018).

The original transformer architecture consists of multiple encoder and decoder blocks, where the output of a block is the input for the next block. The decoder blocks are used to convert embeddings back to words, such as for translation tasks. Bidirectional Encoder Representations from Transformers (BERT), introduced by Devlin et al. (2018), found that, when the decoders are discarded, the final embeddings from the encoder stack can be used for many other applications such as classification and span extraction. Furthermore, they show that BERT can be used for transfer learning, resulting in a single model that outperformed many task-specific models.

1.1.1 Tokenization

When using language models, text needs to be mapped to a numeric input. Most often, a set of frequently used words is assigned a unique number (id). The set of mappings is called a vocabulary, and every item a *token*. A sentence is converted into a vector of token ids, and supplied to the model. Typically, this vector has a maximum length; 512 tokens in the case of BERT. When a batch size larger than 1 is used, "[PAD]" tokens are added to the shorter sentences, such that all vectors are of equal length, and the input matrix is square.

BERT makes use of the WordPiece tokenizer (Schuster and Nakajima, 2012). Words that are unknown to the tokenizer are split into the largest tokens available. For example, if the word "infeasible" is unknown, it can be split up in known tokens "in" and "feasible". The vocabulary consists of 30,000 tokens, including a set of all single symbols, numbers, and characters. Any input can be split up into its characters, and no "unknown" token is needed for words missing from the vocabulary, which can therefore be kept relatively small. Instead, unknown words are learned internally as token combinations by BERT, typically in the lower layers. In case of the running example, it is tokenized as:

"[CLS] one relentless ##ly de ##pressing situation [SEP]"

The "[CLS]" token is added to the start of any sequence, and is used for classification tasks. The "[SEP]" token represents the end of a sequence segment. For example, in question answering, it is used to mark the separation between the question and the answer. If a word is split up, the extra tokens have two hashtags (##) to indicate they are combined with the token before. Note that there are always two hashtags, regardless of the actual number of removed characters.

Therefore, in practice, masked language modeling (MLM) and augmentation occurs at the token level - rather than at the word level. Note that the examples in this work are on word level for *demonstrative purposes*.

1.1.2 Transfer Learning

In transfer learning, a model is first *pre-trained* on one or multiple related tasks, before being *fine-tuned* on the task of interest. A related task may be a related dataset and/or objective. BERT is pre-trained on large bodies of text (corpora), such as Wikipedia, using a masked language modeling task. In MLM, words in a sentence are randomly masked, and the model predicts the missing words. In our example, this would be:

"one relentlessly depressing [MASK]"

The MLM task is used because it allows for pre-training on any corpus, as no additional labels are required. Figure 1.1 shows the general architecture of a BERT model. BERT outputs an embedding for each token. Different objectives require different *heads* that are connected to the output of BERT. For example, the beginning of each sentence is the "[CLS]" token, and its embedding alone is used as the input for classification tasks. During fine-tuning, the embedding of the [CLS] token is trained further, but the other token embeddings are ignored. In masked language modeling, for every masked token independently, the embedding is the input for a classification task over all tokens in the vocabulary.

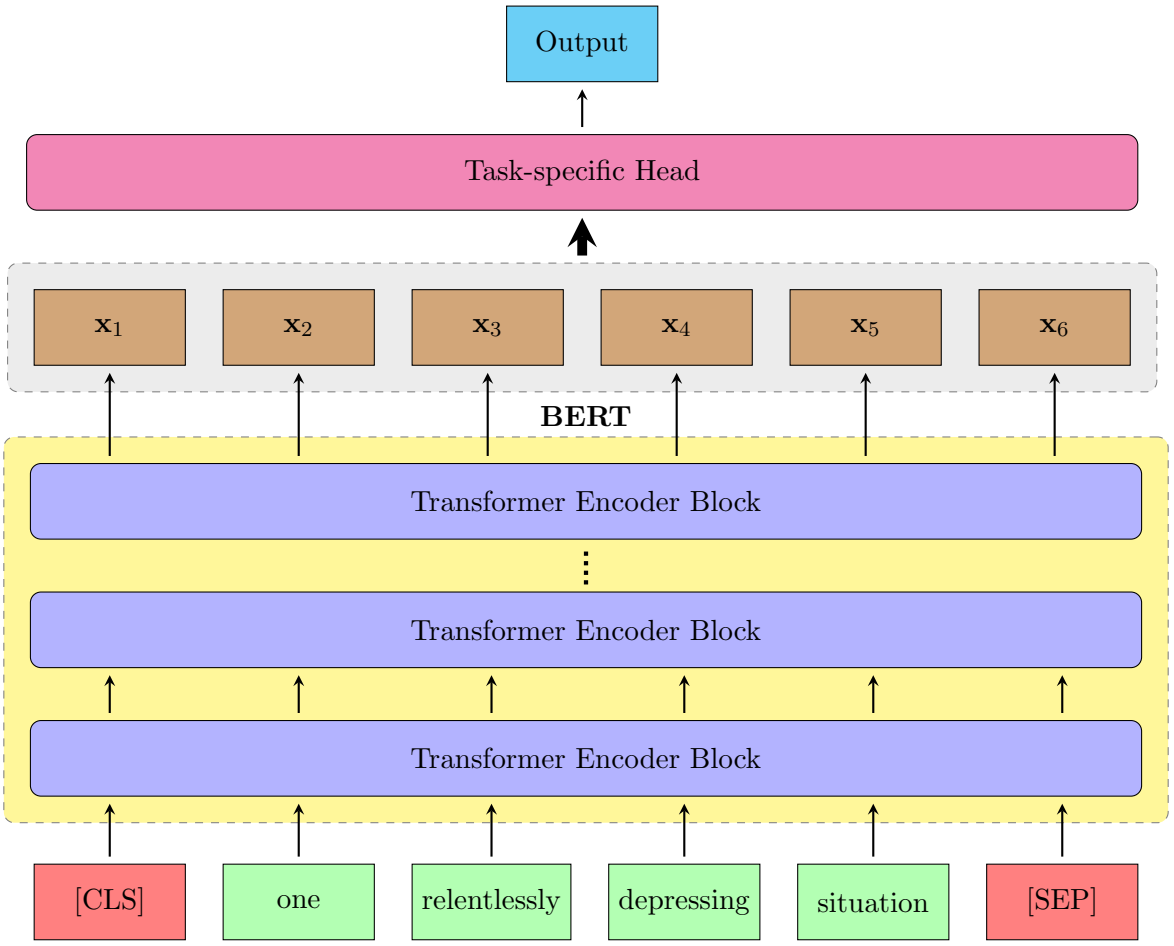

Figure 1.1: General BERT architecture, using the running example as input. $\mathbf{x}_i$ represents the embedding vector as output of BERT for the token at position i. On top a generic task head is attached for illustration.

The benefit of pre-training to the downstream task comes from the weights in the BERT layer having improved over random initialization. Intuitively, one can consider the MLM task on a large corpus, as done by Devlin et al. (2018), as teaching BERT "language", before fine-tuning.

This initial pre-training phase requires a lot of computational resources, which are typically only available to large organizations. By taking the publicly available pre-trained version as starting point, BERT can be fine-tuned for an application with a small dataset in only a few epochs using moderate hardware, obtaining state-of-the-art performance. In this work, we propose additional pre-training tasks using weak supervision to improve an augmentation model that uses BERT.

1.2 Data Augmentation

In computer vision, augmentation operations are often trivial and intuitive. An image can be flipped, cropped, or manipulated otherwise, and still agreeably show the same object. The same does not hold for text. As an example, we consider the negative movie review:

"one relentlessly depressing situation"

Cropping this sentence would give *"one relentlessly"* and flipping it would give *"noitautis gnisserped ylsseltneler eno"*. This shows that these "simple" augmentations can result in losing all meaning of the original text.

To preserve semantically valid sentences, most methods inject or replace words to augment the text. The challenge becomes choosing the optimal words that maintain label quality, while introducing enough diversity for the augmentation to improve generalization. Crucially, the word choice needs to be *conditional* on the label of the sample. A small part-of-speech (POS) change such as:

*"**a** relentlessly depressing situation"*

will only marginally effect the model, acting more as a duplicate. Replacing with a word that is semantically feasible, but ignores the label, can harm the meaning of the sentence, in our example:

*"one relentlessly **brilliant** situation"*

This would completely negate the sentiment of the review. Transformer models are normally fine-tuned on a different type of downstream task, such as classification or regression, using the MLM task for transfer learning only. However, MLM pre-training makes these models ideal candidates for word replacements. Previously, in Wei and Zou (2019), a thesaurus such as WordNet (Miller, 1995) would be used, which might only be partially applicable to the domain, has no probabilistic measure of similarity, and is infeasible to use conditional on a label. BERT, on the other hand, can simply

be trained on a large unlabeled corpus, using the MLM task to learn probable replacements for the specific domain. The question becomes: how to train the BERT augmentation model conditional on the labels? Kumar et al. (2020) found that, for autoencoder models, the most effective and simple way is to train the model using the MLM task on the labeled dataset, and to simply *prepend* the label in natural form as follows:

*"**negative** one relentlessly [MASK] situation"*

where the label is "negative". In this manner, during training, replacement candidates are conditioned on the label. Transformer models have hundreds of millions of parameters, and tend to over-fit quickly. Data augmentation is often used for regularization and increased performance through better generalization by adding variance rather than the potential direct performance gain of incorporating more data.

Data augmentation can be applied both on text or the vector representation (embedding). Embedding augmentation methods insert noise and randomness directly, but lose a trivial mapping to tokens, and therefore, the interpretability of human readable text. In this paper, we choose to limit the scope to text augmentation methods.

1.3 Weak Supervision

Weak supervision aims to obtain low quality labels for the unlabeled data without using the labeled data available. The obtained dataset is used for further pre-training, or even as the only training set. Weak supervision methods, such as Snorkel (Ratner et al., 2020), make use of a combination of expert-defined heuristics, existing models, and any other sources of useful information to estimate training labels without any access to ground truth data.

In Snorkel, the weak supervision model, which consists of the information sources, is called a *generative model*. Next, a *discriminative model* - often a deep neural network - is trained, using the generative model predictions as labels, with a noise-aware loss function to appropriately weigh each observation. Ideally, the discriminative model generalizes beyond the heuristics of the generative model. For example, a heuristic might be a list of negative words that contains the word *"depressing"* but misses the word *"hopeless"*. When using BERT as the discriminative model, both words have similar meaning in the context to a BERT model from pre-training. It can generalize beyond the heuristic and also correctly classify the sentence:

*"one relentlessly **hopeless** situation"*

Essentially, each source of information provides an estimate for the label of each observation. Typically, the sources are assigned reliability weights, using maximum likelihood data programming (Ratner et al., 2016). Next, these estimates are combined into a single estimate, that is the weak label. The sources of information may have different accuracies, which are estimated by Snorkel without ground truth, and may also *abstain* from predicting for a specific observation. To incorporate the different levels of confidence, Snorkel yields *probabilistic labels* rather than binary predictions, meaning that each class gets assigned a probability. Snorkel aims to have the probabilities best reflect the confidence in the labels, rather than minimizing cross-entropy. Labels with less confidence have a lower probability, scaled to act as sample weights. In this manner, labels can have heterogeneous noise levels.

In our research, we assume a Snorkel-like weak supervision method is used. To the best of our knowledge, the generative model has not been used to train an augmentation model in academic research. As data augmentation and weak supervision aim to solve a different part of the same problem, we propose to combine them into a single method.

1.4 Span Extraction

In question answering tasks, a question and a sequence of text containing the answer are given. The model has to highlight only the part of the sequence that is the answer to the question. Such a task is categorized as a span extraction - also referred to as span selection - problem. The problem is formulated as a classification problem over all tokens in the sequence. Typically, there are two classification heads; one to predict the first token in the span, and the other for the last token. Keskar et al. (2019) propose a method to reformulate any task as a span extraction problem by posing a natural question, such that a wider variety of tasks and datasets can be used for transfer learning. In case of the example, the classification task is to determine whether the review is positive or negative:

*"**positive***	*or*	*__negative__*	*?*	*one*	*relentlessly*	*depressing*	*situation"*
0	*0*	*1*	*0*	*0*	*0*	*0*	*0*

Below the tokens, the labels are given. As the review is negative, it is the only token with its label equal to 1. During the conditional augmentation training, the true label needs to be prepended to the sentence in word form. However, when using weak supervision, there is uncertainty in the labels, and prepending the wrong textual label would only harm the training of the augmenation model (AM). We propose to reformulate the classification of the weakly-labeled data, as well as the

expert-labeled data as a span extraction task, such that both textual labels can be included. In this manner, the weakly-labeled data can be used to pre-train the augmentation model not only by training a classifier on domain-specific data, but also by increasing the textual relationship between the token labels and the true meaning of the text, as is needed for the conditional augmentation.

1.5 Contribution

We present a methodology to make use of weak supervision, in order to improve upon data augmentation. We propose new pre-training tasks for the augmentation model, using the weakly-labeled *data*, and extend the augmentation process itself, using weak supervision *methods* as well as introducing our own. Our contributions to the field can be divided into two main categories:

- Augmentation model pre-training through weak supervision and span extraction;
- Heterogeneous augmentation.

For the training of the augmentation model, we propose to pre-train on the weakly-labeled data, using a sequential transfer learning setup, consisting of an MLM task and a noise-aware classification task formulated as span extraction. We use the classification task not only to pre-train the augmentation model, but to *analyze* the difficulty of classifying each data point, which is then incorporated in the degree of augmentation. As we are aware of the extent to which samples have been augmented, we propose a method for estimating probabilistic labels. We coin the observation dependent strategy: *heterogeneous augmentation*. A birds-eye view of the steps in the process is depicted in Figure 1.2.

An extensive set of experiments is done to investigate each component of the proposed methodology ceteris paribus. All results are compared to the work of Kumar et al. (2020), as it is currently considered the state-of-the-art. Furthermore, in most work on data augmentation, the benchmark datasets used are for sentiment analysis or topic classification tasks. These tasks are prime candidates for data augmentation, but also arguably the least challenging. We add two new, complex natural language understanding (NLU) tasks from the GLUE benchmark (Wang et al., 2018) and show that data augmentation is also effective for NLU. We outperform the benchmark for all tasks with up to 4.4 absolute accuracy points, demonstrating the benefit of weak supervision in data augmentation.

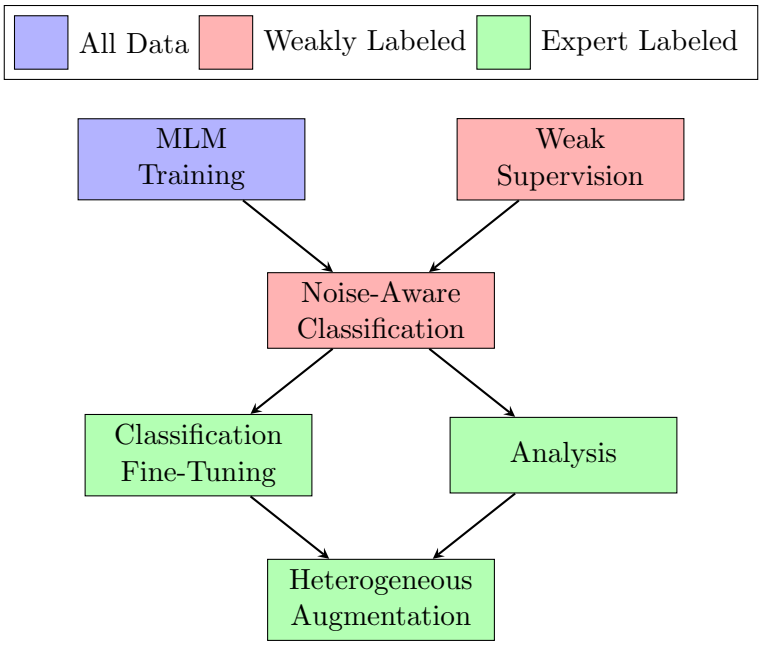

Figure 1.2: Overview of the tasks in the methodology. All steps are color-coded to indicate the data used. The analysis step is an inference phase in which predictions are made for the expert-labeled data, such that the errors can be taken into account for the degree of augmentation, making it heterogeneous.

2 Related Work

In this section, we expand on the field of NLP and the developments that have led to the creation and rise of BERT-like models. Related data augmentation research is discussed in order to provide context to the methods and choices in this paper. Finally, more background information on Snorkel as a weak supervision method is given, as our work assumes that such a system has been used.

2.1 Transformers

Since 2017, transformer models have drastically changed the field of NLP. Previously, the process of building a task-specific language model would consist of vocabulary construction, feature extraction, model design and - if available for the application - pre-trained embeddings such as word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014). The model architecture is typically a logistic regression or recurrent neural network (RNN). In word2vec, a word has a fixed numerical representation or embedding, independent of the other words in a sentence. For example, the word "date" has a single embedding, however, in the sentences:

*"The **date** of the appointment"*

*"He asked her out on a **date**"*

*"My favorite fruit is a **date**"*

the word has different meanings. RNNs are used to incorporate *context* - the other words in a sequence - into the representation of a word. RNNs however, due to their sequential nature, are not parallelizable computationally. Furthermore, the hidden state embedding is calculated using the previous hidden state and the next word, resulting in a vanishing gradient problem. The RNN has been extended to overcome the vanishing gradient problem by Hochreiter and Schmidhuber (1997) who presented the Long Short-Term Memory network (LSTM). The LSTM is more complex, yet still sequential and thus, even more computationally expensive. Language models would typically compute embeddings by reading a sentence only from left-to-right or right-to-left, missing context that might be relevant to the word at hand. Peters et al. (2018) presented ELMo, a shallow bidirectional LSTM architecture, to take all context in a sentence into account.

Google Brain's Vaswani et al. (2017) argued that context can be incorporated without recurring loops and intermediate hidden states, and presented the concept of *self-attention* and the *transformer* architecture. This innovation led the way to many new language models, rendering the previous state-of-the-art obsolete. Most transformer pre-training tasks are formulated, such that no expert labels are needed, and instead, raw text can be used of which there is an abundance on the Web.

OpenAI's GPT-2 (Radford et al., 2019) is an autoregressive (AR) transformer model. AR models are pre-trained using a next word prediction task on a large corpus such as Wikipedia. GPT-2 showed that increasing the amount of transformer layers, and an extensive pre-training phase, generalize well to most downstream tasks. The next word prediction task is used for text generation and translation, because during inference, the target sentence does not exist yet. In our example, this might look as follows:

"one relentlessly [MASK]"

where iteratively new tokens are predicted until a stop token is generated by the model.

However, in many applications, such as classification and question answering, the full target sentence is available at inference. Google AI Language's Devlin et al. (2018) introduced Bidirectional Encoder Representations from Transformers (BERT), which relaxes the next word prediction task in GPT-2, and instead, uses the hidden representation of the final transformer encoder layer

as embedding for a wide range of tasks. BERT is trained using the masked language modeling task also known as the *cloze* task (Taylor, 1953), where a word in the sentence is masked and predicted with the rest of the sentence available. With the same example, the masking task now looks as:

"one relentlessly [MASK] situation"

providing more context, and therefore improved performance for the masked task as well as any downstream tasks that do not involve iterative text generation. Also, a next sentence prediction (NSP) task is included, in which the model has to predict if two sentences from the same document directly follow each other or are sampled randomly. The MLM task allows for deep bidirectional self-attention, as all words of the sentence before and after the target may be considered, and thus, yields more contextual word embeddings.

BERT is pre-trained on large corpora, after which a relatively small dataset for the downstream task is needed, and fine-tuning the model only takes about 2-5 epochs. BERT, as a generalized language model with some fine-tuning, has outperformed task-specific models without any customization to the architecture, vocabulary or training procedure. The pre-training phase is computationally, and thus, financially expensive, making it only feasible for larger institutions. However, Devlin et al. (2018) show that a pre-trained BERT model can be fine-tuned for many domain-specific tasks in a cost-effective manner, as the learned embeddings from a big general corpus can be used for transfer learning, wherefore one does not have to teach a model "language" from scratch.

BERT started a new phase in NLP, in which transfer learning is both practical and the best-performing, similar to the impact of ImageNet (Deng et al., 2009) on computer vision. After BERT, attention-based models with extra complexity started outperforming BERT. However, Facebook AI's Liu et al. (2019b) performed an extensive ablation study of BERT, finding that BERT was under-trained and introduced many small tweaks resulting in a Robustly Optimized BERT pre-training approach (RoBERTa), which again became the state-of-the-art. They argue that the gains made by more complex models should be attributed to better training procedures, rather than the introduced novelties.

The same team of RoBERTa also introduced SpanBERT (Joshi et al., 2020), in which they experiment with randomly selecting spans of text instead of words to mask, as well as adding a span boundary task. Both tasks intend to challenge the model to better learn contextual relations and span extraction tasks. Salesforce Research's Keskar et al. (2019) showed how question answering,

classification, and regression tasks can all be formulated as span extraction tasks, and the transfer learning improvements in this formulation. In GPT-3 (Brown et al., 2020), the model is enlarged to 175 billion parameters, and trained on 570GB of compressed and filtered Web pages, with the aim to be able to pose natural questions to the model, similar to a span extraction task.

The pre-training phase has clear benefits on both the performance and training time of transformer models. The pre-training is usually done using general corpora such as Wikipedia or Common Crawl, before fine-tuning on the domain-specific task with limited data. Phang et al. (2018) found with Sentence Encoders on Supplementary Training on Intermediate Labeled-data Tasks (STILTs) that, if there are other corpora available closer to the end task, it is beneficial to first pre-train further on those corpora before fine-tuning on the downstream task. Microsoft Research's Liu et al. (2019a) present Multi-Task Deep Neural Networks (MT-DNN) for Natural Language Understanding, in which the model is trained on multiple tasks simultaneously rather than sequentially. They found that, aside from the performance benefits, only a very small dataset - of sometimes less than 100 observations - is needed to achieve reasonable results for a downstream task.

The rise of transformers has also resulted in promising advancements in other fields. For example, the same team of the original transformer also proposed Image Transformer (Parmar et al., 2018), significantly improving over the state-of-the-art on ImageNet. In time series forecasting, The Temporal Fusion Transformer (Lim et al., 2019) obtains new state-of-the-art performance in interpretable multi-horizon forecasting.

2.2 Benchmarks

To evaluate general-purpose language understanding models, Wang et al. (2018) introduced GLUE, a collection of existing datasets with (reformulated) tasks, such as question answering, sentiment analysis, and textual entailment. The tasks require a model to capture multi-disciplinary complexities and more human-like reasoning abilities. Most notably, GLUE contains the QNLI task, which is a reformulation using the dataset from SQuAD task (Rajpurkar et al., 2016) - an often used span extraction benchmark.

GLUE's public leaderboard[1] presented a challenge that encouraged competition between universities and corporations, and - as of 2019 - the transformer models are outperforming the non-expert human baselines used for GLUE. To further challenge the status quo, Wang et al. (2019) introduced SuperGLUE, coined the "stickier" benchmark. SuperGLUE consists of more challenging

[1] `https://gluebenchmark.com/leaderboard`

and diverse tasks, often using longer passages of text, whereas GLUE mostly consisted of single sentences.

2.3 Data Augmentation

Often, one has an abundance of unlabeled data but only a small set of high-quality, ground truth labels. One way to make the most use of the data available is to augment existing labeled data into new labeled data. In computer vision, augmentation has proven most useful (Simard et al., 1998). An image can be transformed and still resemble its original. In language, however, transformations that preserve meaning are less trivial. Mixup (Zhang et al., 2018) is an augmentation technique in computer vision that blends training samples. Guo et al. (2019) extend this approach to blending sentences on embedding level, showing minor improvements. Embedding augmentation, however, does not yield human-readable text as samples, thus serving only as a regularization method.

Text augmentation techniques augment the text of the training samples, creating new human-readable samples with two main goals: to enlarge the training set with plausible samples, and as a regularization method improving robustness. HazyResearch[2] is a research group led by Christopher Ré at Stanford, specialized in weak supervision. Hazy Research's Dao et al. (2019) provide a general theoretical framework of augmentation as a Markov Process. They give a theoretical proof that data augmentation increases invariance by feature-averaging, and the variance of the augmented samples acts as a regularization term that penalizes model complexity. Wei and Zou (2019) present Easy Data Augmentation (EDA), a method that augments samples using random swaps, deletions and synonym replacement with varying results.

Kobayashi (2018) first made use of BERT and masked language modeling for augmentation. The aim is to best-create distantly similar plausible samples, while preserving the labels. The current state-of-the-art augmentation processes are conditional on the desired label. CBERT (Wu et al., 2019) adapts the segment embedding of BERT to act as an indicator for the label. This approach is BERT-specific and dependent on the number of labels. Alexa AI's Kumar et al. (2020) show that the label can simply be prepended to the sequence and is model-agnostic. To the best of our knowledge, this is currently the best method.

Quteineh et al. (2020) use data augmentation in combination with active learning and Monte Carlo Tree Search for data generation. They address the situation in which *unlabeled* data is scarce, and domain-experts label generated samples instead. However, one could question the use

[2]http://hazyresearch.stanford.edu

of a model in this scenario, as a model is typically used to limit manual labour when the size of the data scales. If there is little real-world data of interest, and domain-experts are affordable, they could annotate the real data directly instead of generated data.

Jin et al. (2020) investigate BERT's robustness to adversarial attacks, in which the minimal amount of words are replaced with close synonyms, such that the meaning of the sentence is preserved, but the target model starts incorrectly predicting the sample. In this case, augmentation is used for robustness rather than creating more training data.

2.4 Weak Supervision

Another approach to handling the small expert-labeled dataset problem is making use of the vast amount of unlabeled data available. By using external knowledge bases or domain-expert defined heuristics, weak labels can be obtained for the unlabeled data with varying levels of confidence. This confidence can be incorporated with probabilistic labels. Natarajan et al. (2013) present a theoretical experiment for learning with noisy labels.

HazyResearch most notably introduced Snorkel (Ratner et al., 2020), a weak supervision method that requires no human annotated data for the training phase. Instead, a domain expert creates labeling functions that look at specific traits of a sample and vote on the label. Snorkel uses a generative model based on data programming and maximum likelihood (Ratner et al., 2016) to estimate accuracies for the labeling functions without ground truth labels. The weakly labeled dataset is used as training data for a discriminative model, such as BERT, to internalize and improve beyond the labeling heuristics.

Snuba (Varma and Ré, 2018) aims to automate the generation of labeling functions, when a small set of ground truth labels is available. Snuba is primarily effective for computer vision, where manually setting numerical thresholds is tedious. Snorkel MeTal (Ratner et al., 2018) is a multi-task adaption of Snorkel, in which a BERT model uses many different tasks and datasets to overcome specific structural weaknesses on the Recognizing Textual Entailment (RTE) GLUE task.

3 Methodology

In this section, a detailed description of the methodology is given. First, an overview of the entire process is introduced. Next, all steps and choices are described. In the last section, an end-to-end example is included. An overview of the symbols used in this section is included in Appendix A.

3.1 Overview

In data augmentation, the aim is to improve any model of choice, also referred to as the end-model or downstream model, by training it using a high-quality augmented dataset. The augmentation model predicts masked words in a sequence containing the prepended label. Recall the example from Section 1.2, for which the augmentation task is formulated as:

*"**negative** one relentlessly [MASK] situation"*

There is a trade-off between the diversity and quality of an augmented observation. If *"depressing"* is replaced by *"depressive"*, only a small change is made, so there is a high confidence that the label is still valid and its quality is high. However, because this is only a small change, little diversity, and thus, noise, is introduced into the dataset. If instead, a random word, such as *"potato"* would be used, more diversity is added to the dataset, but the quality of the label decreases. Ideally, a word such as *"boring"* is chosen, which both preserves the quality of the label and introduces a new example to the dataset. The quality and diversity can be controlled by the selection strategy of the candidate tokens. By *training* a better augmentation model, the probability distribution of candidate tokens for a masked word more accurately reflects semantic feasibility.

To improve the augmentation model, we introduce a methodology consisting of new and adjusted steps to obtain the augmented dataset. A sequential transfer learning procedure is used containing multiple tasks that improve the quality of the augmentation model. Figure 3.1 shows the flow of the procedure. The tasks include SpanBERT - an MLM task - to train semantically sound word replacement, (weakly) supervised span-extractive (SpEx) classification tasks to train the co-occurrence relations between words and labels, and conditional augmentation model training. The procedure requires a large, weakly-labeled dataset and a small, high-quality labeled dataset. The high-quality dataset holds the observations, which are to be augmented, whereas the weakly-labeled dataset serves to pre-train the augmentation model.

Note that for each step, the training or inference is done on *all* applicable data at once, the steps are *not* executed per observation. Recall Figure 1.1, for each step, the task-specific head is changed, and the improvement of the augmentation model comes from the further pre-training of the weights in the BERT layer only.

The weakly supervised dataset and the span extraction formulation make it *possible* to have more domain-specific pre-training and improved conditional, heterogeneous augmentation. In the current benchmark, Kumar et al. (2020), only the augmentation training (step 4), and augmentation

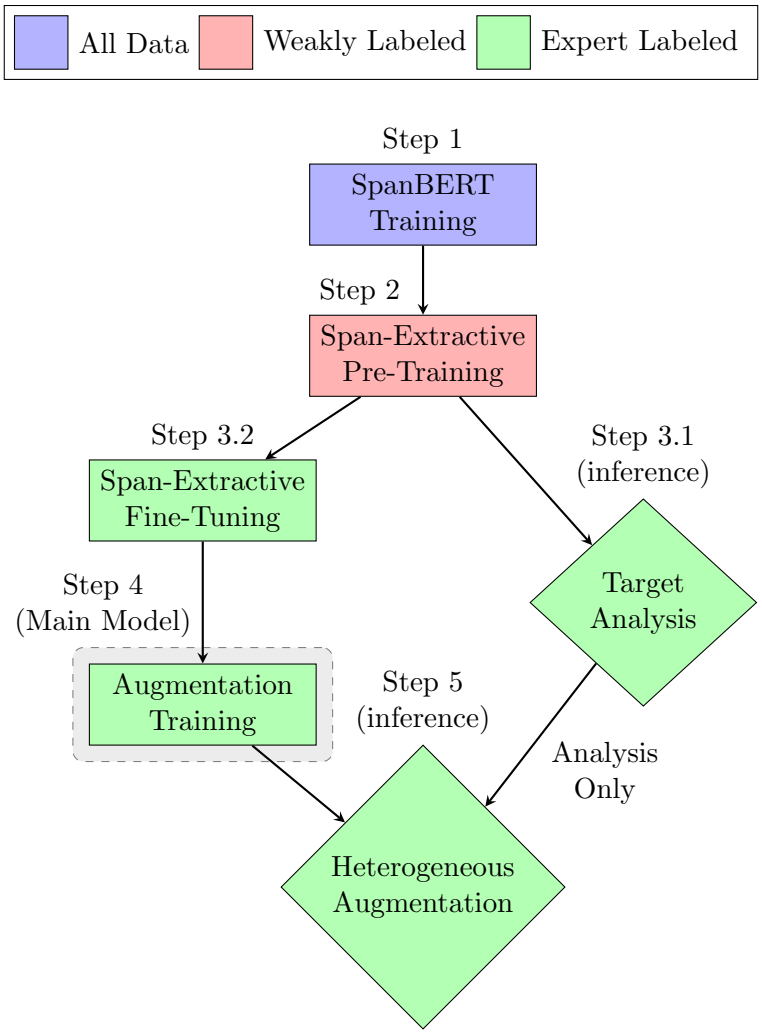

Figure 3.1: Overview of the steps in the methodology. The arrows represent the flow of the augmentation *model*, with the exception of the target analysis, where the complexity and attention of the expert-labeled data is passed. The squares represent training steps, and the diamond shapes represent inference steps. All steps are color coded to indicate the data used.

without target analysis (step 5 adjusted) are present. In the next sections, we describe each step in detail. After the augmented dataset is obtained, it is combined with the original labeled dataset to form the final training set for an end-model of choice.

3.1.1 Implementation Details

The starting point for the augmentation model is a BERT-base uncased, with $L = 12$ transformer blocks, $H = 768$ hidden size, and $A = 12$ attention heads, resulting in 110M parameters. This configuration is chosen as it the most commonly used in the field, mainly because the larger version of BERT does not fit on most GPUs and smaller versions have only been recently introduced. We

make use of the implementation from Huggingface[3], a library providing a common interface for all transformer-based models. We use the original model by Devlin et al. (2018), pre-trained on the BookCorpus dataset and the English version of Wikipedia. Our implementation is in TensorFlow. We make use of layerwise learning rates by using Layer-wise Adaptive Moments (LAMB) as optimizer. Proposed by You et al. (2020), LAMB is originally intended to speed up pre-training by allowing for larger batch sizes without loss in performance. However, You et al. (2020) found that LAMB also yields excellent performance for smaller batch sizes and is typically more consistent than the often used Adam with Weight Decay (Loshchilov and Hutter, 2018).

During training, we make use of *smart batching*. Attention is computed for every token in relation to every other token. Thus, including more tokens increases the number of relations exponentially. Within a batch, all sequences need to be padded to the same length such that they can be fitted into an non-ragged tensor. However, batches do not have to be the same shape. By first sorting the dataset based on string length, and shuffling locally within a range of 3-6 batch sizes as a rolling window to maintain randomness, the maximum sequence length per batch is optimized and computation time is decreased. After the batches have been created, they are shuffled for the training order. Smart batching is especially useful in a dataset with strongly heterogeneous sequence lengths, such as movie reviews, where one can leave a single word or an extensive essay. Decreasing the overall maximum sequence length results in a loss of information, while keeping the maximum sequence length larger results in many unnecessary computation for short reviews. The code is published on GitHub[4].

3.2 SpanBERT MLM

The MLM task is included to both improve augmentation and classification performance. In pre-training, BERT predicts the masked tokens in a sequence. 15% of the tokens in a sequence are randomly selected. Of the selected tokens, 80% is masked, 10% is kept unchanged, and 10% is replaced by a random token. The unchanged set is kept such that the original tokens for the selection remain the most probable. In BERT, pre-training MLM is used to learn embeddings of the corpus and the actual performance is not of importance. For augmentation, however, the performance does influence the quality of the augmentations, although there still may be multiple valid candidate words.

[3] `https://huggingface.co/transformers/`
[4] `https://github.com/timellemeet/dawson`

MLM is a classification task, using a fully connected layer to map the embedding of a token to all tokens in the vocabulary. A visual representation of the MLM task is given by Figure 3.2. Note that the classification head is *shared* across all tokens, meaning that the token classifier has a single set of trainable weights to map an input embedding to a token and is *not* different for each position.

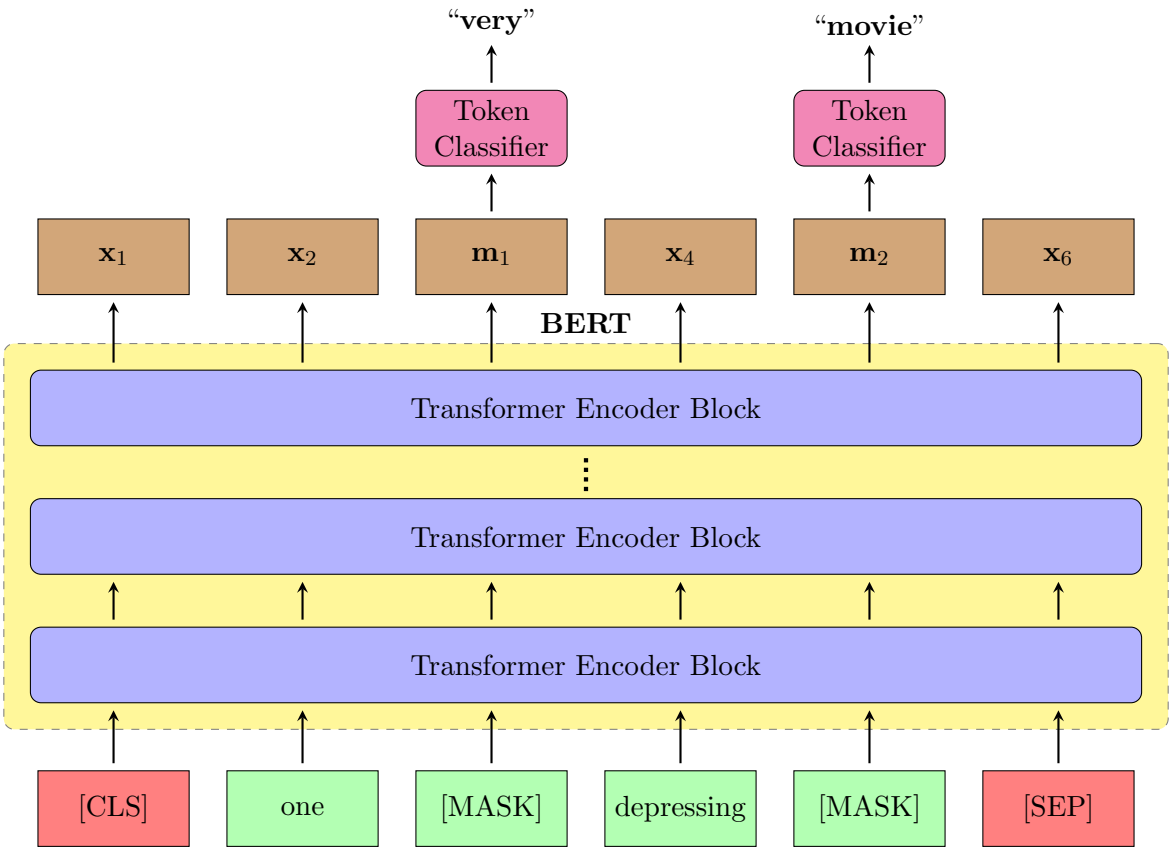

Figure 3.2: BERT architecture for masked language modeling, using the running example as input. The token classifier is identical for each mask. $\mathbf{m}_k$ denotes the embedding for the k-th masked token.

Let $X = \{x_1, \cdots, x_N\}$ be a non-masked sequence of tokens, $\Pi = \{\pi_1, \cdots, \pi_K\}$ the indices of tokens to be masked in sequence X, and $M = \{m_1, \cdots, m_K\}$ the masked tokens in the altered sequence X. Let $\mathbf{m}_i$ be the final vector representation of token m_i from the shared BERT model, before being passed to the task-specific classification head. The probability of the original token for the masked position is given using a softmax function as:

$$p_{MLM}(m_k = x_{\pi_k}) = \frac{e^{\mathbf{w}_{x_{\pi_k}} \cdot \mathbf{m}_k + b_{x_{\pi_k}}}}{\sum_{v=1}^{V} e^{\mathbf{w}_v \cdot \mathbf{m}_k + b_v}} \tag{3.2.1}$$

where $\mathbf{w}_v$ and b_v are the weights and biases of the dense layer, and V is the vocabulary size.

Furthermore, cross-entropy loss is used:

$$\mathcal{L}_{MLM} = -\sum_{k=1}^{K} \log p_{MLM}(m_k = x_{\pi_k}) \tag{3.2.2}$$

SpanBERT extends the MLM task by masking *spans* of tokens, and introducing a Span Boundary Objective (SBO). Joshi et al. (2020) found that SpanBERT is a more challenging pre-training task that not only improves masked language modeling, but also yields greater gains from pre-training, especially for span extraction tasks. Again, 15% of the tokens are masked. However, the words are selected by an iterative process. First, a span length is sampled from a geometric distribution $l \sim \text{Geo}(p)$. Next, a starting point is uniformly chosen. For example, if the drawn span length and drawn starting point are both 2, the running example is masked as:

"one [MASK] [MASK] situation"

This is repeated until 15% of the tokens has been masked. Like in BERT, 80% is actually masked, one half of the remainder is kept unchanged while the other half is replaced randomly.

The Span Boundary Objective is a second task, in addition to the MLM task. The goal is to predict the tokens in a masked span, using only the tokens at the boundaries and the positions. Joshi et al. (2020) found that the SBO forces the start-, and end-token embeddings of a span to summarize the content of the masked span. They found that the SBO greatly improves downstream span extraction tasks, wherefore we include it in the methodology.

Let (s, e) be the start-, and end-indices of a span. For the Span Boundary Objective, an alternative embedding for the masked token is computed, using only $\mathbf{x}_{s-1}$, $\mathbf{x}_{e+1}$ and the positional embedding of the masked token $\mathbf{p}_{\pi_k}$. The architecture of SpanBERT is shown in Figure 3.3.

The new embedding, $\mathbf{m}'_k$, is calculated using two dense layers, layer normalization (Ba et al., 2016) and GeLu activations (Hendrycks and Gimpel, 2016). The first dense layer takes as input the concatenation of the start-, end-, and positional token, reducing the vector back to the normal hidden size of an embedding. The second dense layer is part of the token classifier, as for MLM. The steps are denoted as:

$$\begin{aligned} \mathbf{h}_0 &= [\mathbf{x}_{s-1}; \mathbf{x}_{e+1}; \mathbf{p}_{\pi_k}] \\ \mathbf{h}_1 &= \text{LayerNorm}(\text{GeLu}(\mathbf{W}_1 \mathbf{h}_0)) \\ \mathbf{m}'_k &= \text{LayerNorm}(\text{GeLu}(\mathbf{W}_2 \mathbf{h}_1)) \end{aligned} \tag{3.2.3}$$

The probability density, and loss function are identical to the MLM task:

$$p_{SBO}(m_k = x_{\pi_k}) = \frac{e^{\mathbf{w}_{x_{\pi_k}} \cdot \mathbf{m}'_k + b_{x_{\pi_k}}}}{\sum_{v=1}^{V} e^{\mathbf{w}_v \cdot \mathbf{m}'_k + b_v}} \tag{3.2.4}$$

$$\mathcal{L}_{SBO} = -\sum_{k=1}^{K} \log p_{SBO}(m_k = x_{\pi_k}) \tag{3.2.5}$$

The final SpanBERT loss is the sum of both the MLM and SBO task losses:

$$\mathcal{L}_{SB} = \mathcal{L}_{MLM} + \mathcal{L}_{SBO} \tag{3.2.6}$$

The SpanBERT pre-training task is done on the full corpus, which includes both the labeled and unlabeled datasets, as no labels are required.

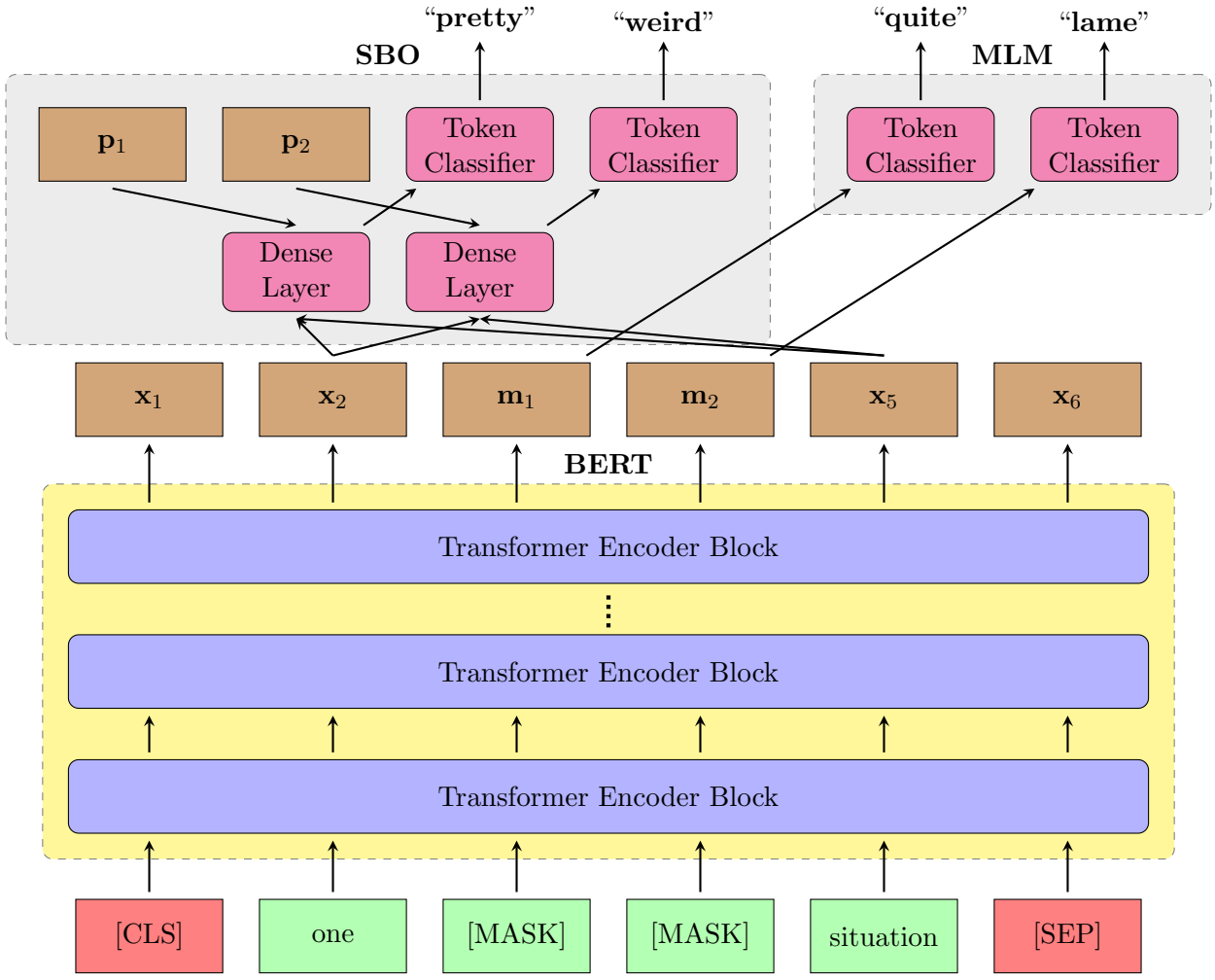

Figure 3.3: SpanBERT architecture for masked language modeling and the Span Boundary Objective, using the running example as input. The token classifier is identical for each mask, but in this case, different per task. $\mathbf{m}_k$ denotes the embedding for the k-th masked token. $\mathbf{x}_2$ and $\mathbf{x}_5$ are the boundary tokens.

In our implementation, since we mask within individual observations rather than a continuous text, we calculate observation-specific parameters such that on average, 15% of an observation is masked. Furthermore, we only have one span per observation for simplicity and we never mask the boundary tokens. During training, the dataset is repeated 10 times, so that the same observation is included with 10 different spans. In this manner, we adjust for only having a single span and make sure that there is a lot of variety in how the model must predict masks in each version of the same observation, and forcing it to generalize more. A validation set is used to prevent over-fitting.

3.3 Span-Extractive Training

The classification task is included to condition the words in a sequence on the label. This way, during the conditional augmentation, the label actively influences the replaced tokens, maintaining sample quality. Since the label is present at the start of the sequence during augmentation, it needs be present during the training of the augmentation model as well. A regular classification architecture would not condition the words on the textual names of the classes. Furthermore, the augmentation model is trained on the weakly-labeled dataset, thus, the labels contain noise and prepending the incorrect label is harmful. Like in weak supervision, probabilistic labels are required to incorporate the confidence of a sample, while conditioning the labels. Keskar et al. (2019) propose a method to reformulate various tasks as span extraction tasks. Both the positive and negative labels are prepended as words, and the objective is to select the span containing the correct label. In Keskar et al. (2019) the labels are included in the form of a natural question:

*"**positive** or **negative** ? one relentlessly depressing situation"*
0 0 1 0 0 0 0 0

We diverge from the original paper by using a noise-aware loss function, not posing a natural question and selecting a single token only, instead of a span, where possible. In order to best mirror the task at the augmentation stage and to reduce complexity, only the labels are included, omitting the tokens needed to phrase a natural question. Suppose that in the example, the weak supervision estimates with 70% probability that the review is negative, the training input is:

*"**positive** **negative** one relentlessly depressing situation"*
0.3 0.7 0 0 0 0

with the labels shown below their respective tokens. Unlike the original formulation, the order of the textual labels is also randomly *shuffled* for each observation, such that the model is forced to train on the actual token-label rather than its position.

In span extraction tasks, there are two trainable parameter vectors, one each for the start-, and end-tokens. However, most simple natural labels - such as *positive* and *negative* in our example - will be present in the vocabulary, and not be split-up in multiple tokens. If this is the case, we propose to simplify the span extraction task to only one trainable parameter vector, the *token selection pointer* $\mathbf{s}$. The probability of token x_i being the true label is given as:

$$p_{SpEx}(y = x_i) = \frac{e^{\mathbf{s} \cdot \mathbf{x}_i}}{\sum_{j=1}^{N} e^{\mathbf{s} \cdot \mathbf{x}_j}} \tag{3.3.1}$$

In Figure 3.4, the span extraction task is visualized. Note that, for the span extraction task, each token's unique embedding $\mathbf{x}_i$ is combined with a single trainable vector $\mathbf{s}$, whereas for the MLM task, the embedding from a single token $\mathbf{m}'_k$ is combined with unique weights $\mathbf{w}_v$ for every target token v. In case the natural label consists of multiple tokens, the implementation remains a standard span extraction task, where two token selection pointer heads are used to predict the start-, and end-token of the label.

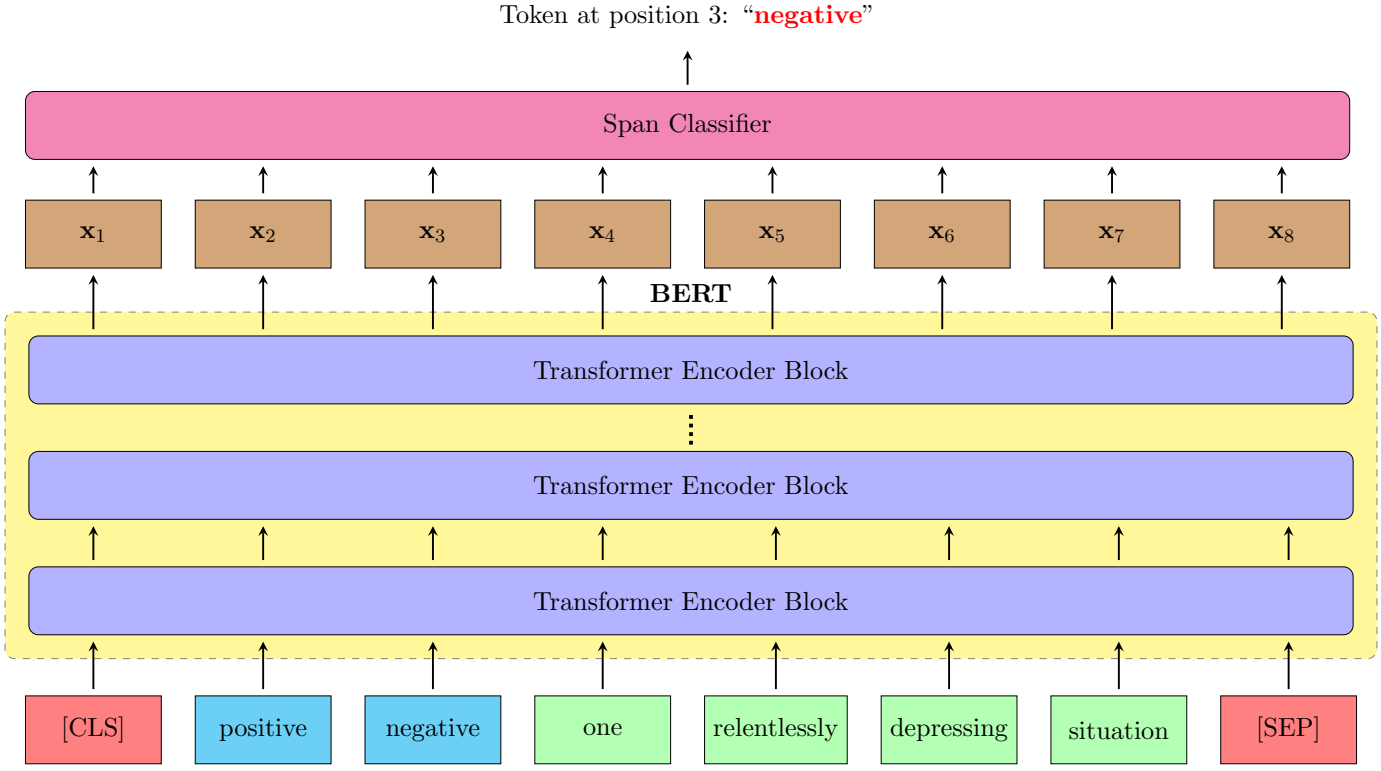

Figure 3.4: BERT architecture for span-extractive classification, using the running example as input. The span classifier does *not* contain a dense layer, but only consists of Equation 3.3.1.

We add a noise-aware loss function to make use of the noise information of the weak supervision. Ground truth labels are unknown, but from the weak supervision phase, probabilistic labels are obtained. A probabilistic label is simply a probability instead of a discrete label. In case there is much uncertainty about a sample, all probabilities will be equal, whereas if there is much confidence in the prediction, the probability for the likely label will be close to 1, and all others close to 0. Let $\tilde{y}$ be the weak label for a sample. For the token selection, we extend the probabilistic label density by including all other tokens:

$$p_{WS}(y = x_i) = \begin{cases} Pr(\tilde{y} = x_i) & \text{if } x_i \text{ label} \\ 0 & \text{if } x_i \text{ not label} \end{cases} \tag{3.3.2}$$

The confidence is incorporated in the loss function to act as sample weight using cross-entropy:

$$\mathcal{L}_{SpEx} = -\sum_{i=1}^{N} p_{WS}(y = x_i) \log p_{SpEx}(y = x_i) \tag{3.3.3}$$

Initially, the model is pre-trained on the large, weakly-labeled data, after which the model is fine-tuned on the expert labeled data. Although the datasets could be merged for a single training step, they are kept separate, such that a target analysis of the labeled data can be done, as well as to ensure that the final fine-tuning is on the highest quality data only. For both the fine-tuning and pre-training, the model is trained for at most 10 epochs, but with an early stopping rule using the development - also know as validation - dataset. Early stopping is a method to prevent over-fitting, where the model stops training once the validation loss starts decreasing, and the weights after the epoch with the lowest validation loss are selected.

3.4 Target Analysis

Samples may have different levels of complexities and the extent to which a sample can be augmented while preserving label quality varies. By including the weakly supervised training step, a classification model for the task is obtained, for which the labeled data is out-of-sample. By predicting for the labeled data, and comparing the results to the ground truth labels, the size of the error gives an estimate for the difficulty of classifying a sample.

$$e = 1 - Pr(\tilde{y} = y) \tag{3.4.1}$$

This is similar to an expected error reduction query strategy in active learning. Furthermore, the relative importance of the tokens to the model is stored using attention. In BERT, there are 12 layers, and for each layer, 12 attention heads. An attention head outputs, for every token, a probability density over all tokens in the sentence. The probabilities act as weights that are used when calculating the output embedding for the token. We take the attentions from the last layer only, and compute the average over all heads and tokens to obtain a final vector or probability density, which is considered as the weights of importance of the tokens. Section 3.5 describes how the heterogeneous augmentation makes use of the error analysis and attention.

3.5 Heterogenous Augmentation

From the target analysis step, information is known about each observation. This can be incorporated in *which* tokens are masked, and *how* they are replaced. The probabilities of the replacement tokens can be used to estimate probabilistic labels. Therefore, we consider this observation-specific augmentation *heterogeneous*.

A sample is augmented by masking a token, prepending the label token, and predicting probable tokens for the mask, conditional on the label. In our example:

*"**negative** one relentlessly [MASK] situation"*

The level of augmentation can be controlled in two directions: the amount of augmented tokens, and the likelihood of the replacement candidates. Again, the amount of masked tokens is kept fixed at 15%. The masked positions are sampled using the *attention vector* from the target analysis instead of a uniform distribution. This selection strategy is more efficient, as the augmented samples contain more information for the classification model.

The augmentation model computes a distribution of probabilities for the token candidates of a masked position. If a sample is complex and already hard to classify, more probable tokens are selected to preserve label quality. Only the expert-labeled dataset is used for both training and augmentation.

3.5.1 Augmentation Training

The augmentation model is fine-tuned on the labeled data itself using the augmentation task. First, the dataset is duplicated 10 times, tokens are randomly masked, and the label prepended. The duplication is used in order to train different masks in the same sentence. The model is trained

for up to 15 epochs, but again with early stopping using the validation dataset to prevent overfitting. Like Kumar et al. (2020), the initial learning rate is kept at $4\epsilon - 5$. The masked language modeling training strategy is the standard BERT task, as described in Section 3.2, but with the labeled prepended as token. Note that the span *masking* strategy and span boundary *task* are omitted, and the masking is uniform instead of using the attention from the target analysis, to train a generalized augmentation model.

3.5.2 Candidate Selection

Depending on the prediction error for a sample during the target analysis, more or less diversity is permitted. A task-specific upper bound (UB) and lower bound (LB) are set empirically for the probability range of eligible replacement tokens. Using the prediction error e_s for observation s, an observation-specific lower bound LB_s is used, defined as:

$$\text{LB}_s = \text{LB} + (\text{UB} - \text{LB})e_s \tag{3.5.1}$$

The tokens in the vocabulary are sorted by probability for each observation, and a token is discarded if the cumulative probability up to-, and including that token is out of bounds. The leftover candidate tokens are re-weighted, using a softmax mapping based on their original probabilities. The resulting probabilities are used to sample the final selected token. By setting the upper and lower bound on the cumulative distribution of candidate tokens, tokens that are not diverse enough or too unlikely, can be omitted. Thus, the overall level of noise can be controlled. As the augmentation model improves through (pre-)training, the probability of suitable tokens increases, while the probability for the rest of the vocabulary decreases, thus, allowing for more diverse sampling while preserving quality.

3.5.3 Probabilistic Labels

In contrast to Kumar et al. (2020) and Wu et al. (2019), we make use of probabilistic labels like in weak supervision. Normally, the *original* binary labels are used. The augmented samples introduce uncertainty and noise, and, as the degree of augmentation is known, an estimation of the reliability of a label can be made. In determining a formulation for the probabilistic label, the following considerations have been made:

- The probabilistic label is a function of token probabilities;
- Adding a token mask should always decrease confidence;
- The label should be roughly in the neighborhood of the lowest token probability;
- The probability of a candidate token is relative to all other tokens in the vocabulary. As the vocabulary is large - and many tokens may be feasible - even the largest token probabilities are typically below 10%;
- The label of the observation may never flip, thus, the confidence is at least 50%.

The probability for the augmented observation label y^* is calculated using average probability for the tokens in the sentence, that is:

$$Pr(y^* = y) = max\left(\frac{N - K + \sum_{k=1}^{K} p(m_k = \hat{x}_{\pi_k})}{N}, 0.50\right) \tag{3.5.2}$$

where $\hat{x}_{\pi_k}$ is the selected replacement token for mask m_k, $p(m_k = \hat{x}_{\pi_k})$ is defined in Equation 3.2.1, and $N - K$ represent the sum of non-augmented tokens having a probability of 100%. Suppose in the example

*"**negative** one relentlessly [MASK] situation"*

with only masked token ($K = 1$) at position π_1, the token *"sad"* is selected as replacement $\hat{x}_{\pi_1}$ and the probability of the replacement is $p(m_k =$ *"sad"*$) = 5\%$. The total number of tokens is $N = 4$, as the label is omitted, as it is removed after the augmentation. The confidence is calculated as:

$$Pr(y^* = \textbf{negative}) = max\left(\frac{4 - 1 + 0.05}{4}, 0.50\right) = 0.7625 \tag{3.5.3}$$

3.6 End-to-End Example

For the step-by-step example, the methodology is applied to two movie reviews. The first review - the running example - is a *negative* review taken from the expert-labeled dataset:

"one relentlessly depressing situation"

The second review is taken from the large unlabeled dataset:

"even if you've never heard of chaplin, you'll still be glued to the screen"

STEP 1: SpanBERT

First, a BERT model, pre-trained by Devlin et al. (2018), is initialized with a SpanBERT head on top. No labels are required, wherefore, all available data can be used. Suppose that for the examples, span lengths of $l = 1$ and $l = 2$ are drawn respectively. The starting point of the spans is random. In the example, the masks are placed as:

"one [MASK] depressing situation"

"even if you've never heard of [MASK] [MASK] still be glued to the screen"

The dataset is repeated 10 times, such that masks are placed at different places in a sentence. The masks are predicted both using the full sequence and just the boundary tokens, as explained in Section 3.2. In this manner, the model is trained for a predefined number of epochs. The SpanBERT pre-training both fits the model to the domain-specific data and improves augmentation.

STEP 2: Span-Extractive Pre-Training

Next, the model is trained on the unlabeled data only, using weak labels. Suppose that the weak supervision method estimates there is a probability of 60% that the unlabeled example is positive, that is $Pr(\tilde{y} = \textbf{positive}) = 0.6$. Both textual labels are prepended and the model is trained using span extraction. The sentence, with probabilistic labels, looks as follows:

*"**positive** **negative** even if you've never heard of chaplin, you'll still be glued to the screen"*
0.6 0.4 0 0 0 0 0 0 0 0 0 0 0 0 0 0

The model has to both determine the sentiment of the sentence, and select the token-label with the corresponding sentiment.

STEP 3.1: Target Analysis

In the previous step, a classifier is trained, that has not yet seen the sentiment of the expert-labeled review. As such, first the classifier is used to make a prediction for the sentence, which can be compared to the true label as analysis of the complexity. Suppose the classifier makes the following prediction:

*"**positive** **negative** one relentlessly depressing situation"*
0.3 0.6 0 0.1 0 0

The predicted probabilities to select a token are shown below the corresponding token. For illustration purposes, the classifier has also incorrectly assigned some probability to the non-label

token "relentlessly". It is known that the correct label is negative, and the model has predicted the negative token is to be selected with 60% probability. Therefore the error - or complexity - is 40%, that is $e = 0.4$.

Not only the prediction, but also the attention to the tokens is saved. The label tokens are removed, and the remaining probabilities re-weighted such that again, their sum is equal to 100%. For the example, the attention probabilities for the remaining tokens are:

"one relentlessly depressing situation"
0.1 0.2 0.4 0.3

STEP 3.2: Span-Extractive Fine-Tuning

After the target analysis, the model is trained further on the expert-labeled data. The fine-tuning procedure is identical to the pre-training, except that, instead of weak labels, the ground truth labels are used:

*"**positive** **negative** one relentlessly depressing situation"*
0 1 0 0 0 0

Note that the span-extractive pre-training step already trained the classifier to only select the label tokens and generalize beyond the noisy labels as much as possible. During the fine-tuning step, only the relationship between the token-label and corresponding sentiment has to be strengthened further.

STEP 4: Augmentation Training

The final training step is the conditional augmentation task. Only high-quality textual labels may be prepended, therefore the model is trained on the expert-labeled data only. The tokens are masked randomly:

*"**negative** one relentlessly depressing [MASK]"*

Because of the pre-training steps, the augmentation model will allocate more attention to the textual label, and make use of the sentiment of the sentence when predicting for the masked token. Like in the SpanBERT step, the dataset is repeated 10 times with different masks.

STEP 5: Heterogeneous Augmentation

The last step is the actual augmentation that is used to create the augmented dataset. Different than in the augmentation training, the dataset is not necessarily repeated, therefore the tokens

are not masked using a uniform distribution, but with the attention probabilities from the target analysis, in order to increase the probability that relevant words are masked. In this case, the token "depressing", with the largest probability, is drawn:

*"**negative** one relentlessly [MASK] situation"*

When selecting a replacement token, a lower, and upper bound are used. We empirically set the general upper bound (UB) to 0.95 and lower bound (LB) to 0.8. Recall the error from the target analysis is 0.4. The observation-specific lower bound is:

$$\text{LB}_s = 0.8 + (0.95 - 0.8) * 0.4 = 0.86 \tag{3.6.1}$$

When predicting a masked token, for each token in the vocabulary, a probability is estimated. We sort the tokens based on their respective probabilities, and calculate the cumulative probability up to each token. The tokens are only considered candidates if the associated cumulative probability is within the bounds of $1 - \text{UB}$ and $1 - \text{LB}_s$. In the table below, the probabilities for all tokens in the vocabulary are given. Only the tokens in between the dotted lines are within bounds.

Table 3.1: Candidate tokens for the masked token in the example.

Token	Probability	Cumulative
Horrible	0.05	0.05
Bad	0.03	0.08
Sad	0.02	0.10
Boring	0.02	0.12
Lame	0.02	0.14
Mediocre	0.01	0.15
⋮	⋮	⋮
Parachute	0.00	1.00
Catalogue	0.00	1.00

For the remaining tokens, their probabilities are re-scaled to again sum up to 100%. In the case of this example, there are 4 candidate tokens. The final token is sampled using the re-scaled probabilities. In this case, the augmented sample is:

*"one relentlessly **boring** situation"*

where the token *"boring"* is selected as replacement. As the number of masked tokens and the probability of the replacements are known, a probabilistic label is calculated to account for the

added uncertainty in the data. The total number of tokens is $N = 4$, the number of replaced tokens is $K = 1$, and the probability of *"boring"* is 0.02. Thus, the probability of the augmented review's label is:

$$Pr(y^* = \textbf{negative}) = max\left(\frac{4 - 1 + 0.02}{4}, 0.50\right) = 0.755 \tag{3.6.2}$$

If the probabilistic labels were to be omitted, $Pr(y^* = \textbf{negative})$ would be equal to 1. The augmented review is added to a new dataset of augmented observations. After all expert-labeled data is augmented, both datasets are combined. Finally, a new classifier - of any kind - may be trained on the combined dataset, using a normal classification formulation.

4 Experimental Setup

The methodology is evaluated on multiple types of binary classification tasks. An ablation study is done to understand the contribution of the different components to the overall performance. In the next sections, the datasets are introduced, the evaluation criteria are determined, and the experimental results are discussed.

4.1 Benchmark Tasks

We make use of a selection of the GLUE tasks (Wang et al., 2018) which form the benchmark for leading NLP models. Examples for the tasks are given in Table 4.1. The GLUE tasks require more non-linear complexity, which is not captured by logistic regression bag-of-words models. Currently, autoencoder transformer models such as RoBERTa represent the state-of-the-art for the benchmark. To the best of our knowledge, it is the first time QQP, QNLI, or other GLUE natural language understanding tasks are used in public data augmentation research.

SST-2

The Stanford Sentiment Treebank (SST-2) (Socher et al., 2013) is a binary sentiment classification task on movie reviews. The review sentiment is human-annotated on the full sequences. Reviews are either labeled positive or negative. The classes are balanced and the GLUE metric is accuracy.

QQP

The Quora Question Pairs (QQP) task (Iyer et al., 2017) consists of pairs of questions, that are classified as semantically equivalent or not. The classes are imbalanced with 63% negative cases and the GLUE metrics used are accuracy and macro-F1.

QNLI

The Stanford Queston Answering Dataset (SQUAd) (Rajpurkar et al., 2016) consists of paragraphs drawn from Wikipedia and annotator-written questions. The task is repurposed as the sentence pair classification task Question-answering NLI (QNLI). Randomly selected sentences from a paragraph are individually paired with their corresponding questions. The task is to classify whether the drawn sentence contains (entails) the answer to the paired question. The metric used is accuracy.

Table 4.1: Examples from the SST-2, QQP & QNLI tasks.

SST-2	Review	*A muddle splashed with bloody beauty as vivid as any Scorsese has ever given us.*
	Label	**Positive**
	Review	*Which half of dragonfly is worse? The part where nothing 's happening , or the part where something 's happening?*
	Label	**Negative**
QQP	Question A	*What is an accurate way to calculate your IQ?*
	Question B	*What's the most accurate way to test my IQ?*
	Label	**Similar**
	Question A	*What is the mechanism of letter of credit?*
	Question B	*How do I get a loan with a letter of credit from another bank?*
	Label	**Different**
QNLI	Question	*When is the term 'German dialects' used in regard to the German language?*
	Answer	*When talking about the German language, the term German dialects is only used for the traditional regional varieties.*
	Label	**Entailed**
	Question	*How many lines are in the rail network?*
	Answer	*Five new light rail lines are currently in various stages of development.*
	Label	**Missing**

4.1.1 Expert-labeled Dataset Selection

The selected datasets are large and therefore suitable candidates for the weak supervision approach, and resemble most practical use cases. For the training and development (validation) datasets, all true labels are available. Not all test sets are publicly available, but used instead for a true out-of-sample comparison on the GLUE leaderboard[5]. For consistency, we fully omit the test sets, and instead, sample our own from the training data.

Table 4.2: The number of observations in the datasets, as well as the mean number of tokens in the observations.

Dataset	Train	Dev.	Test	Mean Tokens
SST-2	67,349	872	1,821	13.3
QQP	363,849	40,430	390,965	30.4
QNLI	104,743	5,463	5,463	50.0

To simulate having a small dataset with high-quality labels, for each iteration of an experiment, two small datasets are sampled from the training data, one serving as the small expert-labeled dataset - that is augmented - and the other as the test set for the experiment. The remaining training data is treated as if it is unlabeled, and a weak supervision method has generated weak labels. The original development sets are used for early stopping, if indicated in the methodology, to ensure a comparable optimization as to any other GLUE based research. For SST-2 and QNLI, the sampled datasets consist of 1% of the observations, whereas for QQP, this is 0.5%.

Table 4.3: The average number observations in the experimental datasets.

Dataset	Weakly Labeled	Expert Labeled	Test
SST-2	66,002	673	673
QQP	360,211	1,819	1,819
QNLI	102,648	1,047	1047

These conditions match most practical use-cases, where collecting ground-truth labels is feasible for evaluation, but not for training large transformer models. The remaining datasets are of sufficient size for most weak supervision methods.

[5] `https://gluebenchmark.com/leaderboard`

4.1.2 Simulating Weak Supervision

To simulate weak supervision, probabilistic labels are generated. The Beta distribution is selected due to its domain of $[0, 1]$ and flexible shape, allowing for different types of noise settings. For interpretability, the α and β parameters are rewritten to the mean and variance of the distribution, which are empirically set by the practitioner. The mean and variance of the Beta distribution are given by:

$$\begin{aligned} \mu &= \frac{\alpha}{\alpha + \beta} \\ \sigma^2 &= \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} < \mu(1 - \mu) \end{aligned} \tag{4.1.1}$$

The definitions of α and β can be reformulated as follows:

$$\begin{aligned} \alpha &= \left(\frac{1 - \mu}{\sigma^2} - \frac{1}{\mu}\right)\mu^2 \\ \beta &= \alpha\left(\frac{1}{\mu} - 1\right) \\ Pr(\tilde{y} = y) &\sim \text{Beta}(\alpha, \beta) \end{aligned} \tag{4.1.2}$$

Most weak supervision methods require the labeling process to be better than random. Therefore the domain of μ is restricted further. The domain of the variance is bounded by the mean as follows:

$$0.5 < \mu < 1 \tag{4.1.3}$$

$$0 < \sigma^2 < \mu(1 - \mu) \tag{4.1.4}$$

The parameters need to be set such that the quality of the simulated weak supervision method is realistic. Matthews correlation coefficient (MCC), proposed by Matthews (1975), is a metric used in binary classification to measure the randomness of the predictions irrespective of class balance. It uses all parts of the confusion matrix and it is symmetrical. MCC is bounded between -1 (perfectly negatively correlated) and 1 (perfectly correlated), with 0 being completely random. We use MCC to evaluate the quality of the generated noisy labels. To simulate a real-life weak supervision scenario for complex tasks, we empirically set $\mu = 0.57$ and $\sigma^2 = 0.05$. Figure 4.1 shows a histogram of draws from the Beta distribution to visualize the generated noise.

The random draws have fat tails, such that a lot of noise is present during pre-training. Table 4.4 shows the MCC and accuracy of the simulated weak supervision method, compared to the true labels that are available for the datasets.

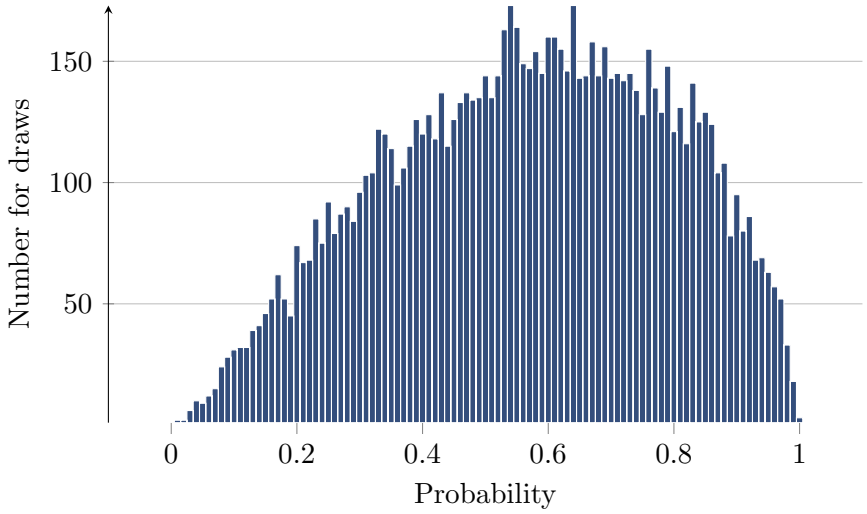

Figure 4.1: Histogram of random draws of a Beta distribution with $\mu = 0.57$ and $\sigma^2 = 0.05$, for the weak supervision simulation. The draws are task-independent. In the histogram, the distribution of 10,000 draws is shown.

Table 4.4: The performance metrics of the simulated weak supervision method. The metrics used are the mean MCC and accuracy of the generated weak labels compared to the ground truth.

	SST-2	QQP	QNLI
MCC	0.244	0.235	0.242
Accuracy	0.623	0.622	0.621

For all datasets, the noisy labels are better than random, and thus, contain information that a discriminative model can generalize. However, the labels are of low enough quality to simulate a weak supervision method.

4.2 Evaluation Criteria

For a direct comparison to the state-of-the-art, we follow Kumar et al. (2020) in the intrinsic and extrinsic evaluation methods.

4.2.1 Intrinsic

The intrinsic evaluation consists of semantic fidelity and generated diversity of the augmented samples. The semantic fidelity is determined by training a BERT-base model on labeled data originally available, with the true labels. Next, the model makes predictions for the augmented dataset, which serve as *ground truth* labels, and are compared to the probabilistic labels from Section 3.5.3. If the difference between both is small, the semantic quality of the augmented

dataset is high. Like in Kumar et al. (2020), the semantic fidelity is measured in accuracy, such that the results can be compared. The semantic fidelity measures to what extent the augmented text and (probabilistic) label correspond.

The generated diversity is measured using the type-token ratio (Roemmele et al., 2017). Diversity is important, as it measures the extent to which the augmentation introduces variability to the existing samples. The type-token ratio is the number of unique predicted tokens (types) divided by all predicted tokens in the dataset. When an MLM model is undertrained, it typically only predicts tokens such as "a" or "the", which would result in a low type-token ratio.

4.2.2 Extrinsic

The extrinsic evaluation is the end-to-end performance. The augmentation model is used to create augmented observations that are added to the original expert-labeled dataset. Next, this *combined* dataset is used to train a classifier (the downstream model), using a *normal* classification task, not a span-extractive formulation. The architecture of a normal BERT classifier is depicted in Figure 4.2. The classifier is evaluated on the test set, which is kept out-of-sample throughout all steps.

Any type of classifier - not only BERT - can be used to evaluate the performance using the improved dataset. We compare two classifiers for the extrinsic evaluation: a newly initialized BERT-base model - only pre-trained by Devlin et al. (2018) - and the augmentation model itself, to make use of the transfer learning from the domain-specific tasks. Both are *fine-tuned* in an identical way with a normal classification head on the combined dataset. Note that this implies that the augmentaion model will train on the samples it has augmented. In the results, the extrinsic evaluation using the newly initialized BERT model, is referred to as *Base*, and the evaluation with the augmentation model as *AM*. Both models have the same architecture with the newly initialized classification head, the *only* difference is the starting point of the *weights* of the BERT layer before fine-tuning.

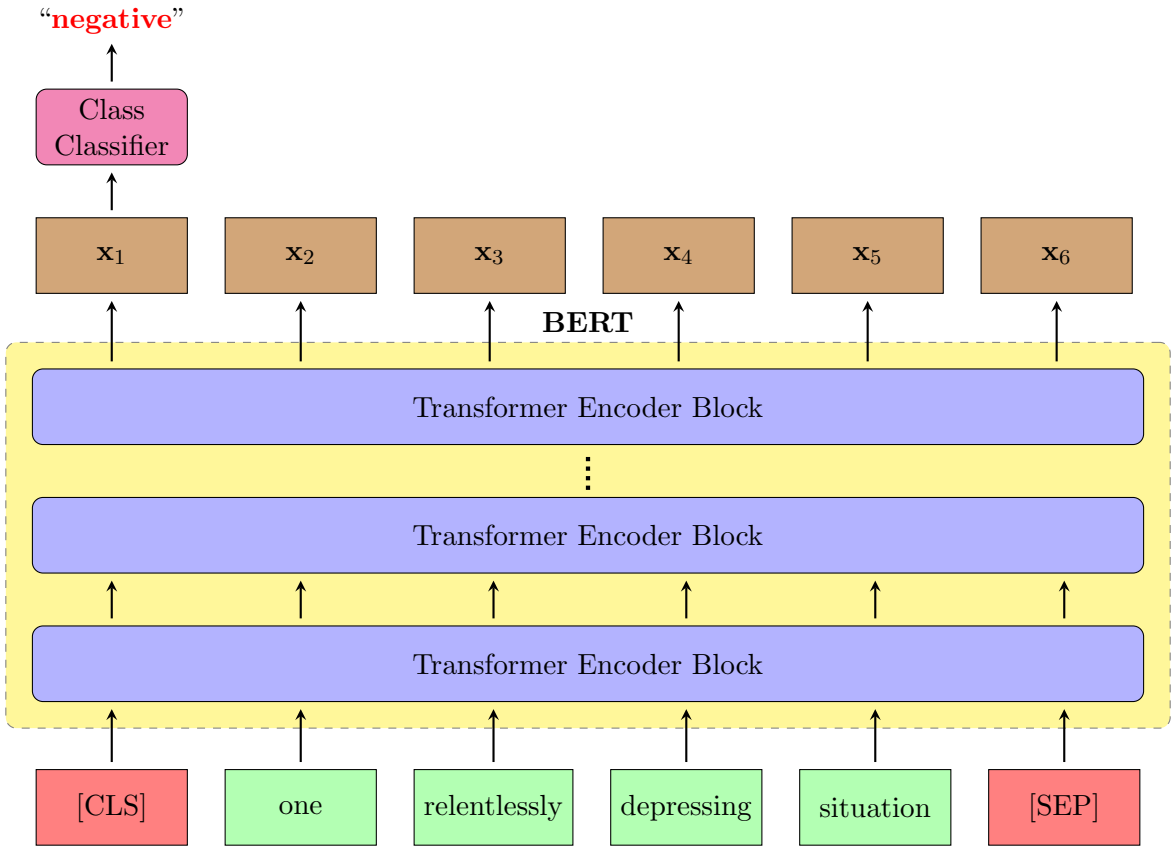

Figure 4.2: BERT for regular classification. The label of the sentence is summarized in the embedding for the "[CLS]" token. The embedding, $\mathbf{x}_1$, is often referred to as the *pooled* output of BERT. The embedding is the input of a single dense layer, with as output dimensions the number of classes, like in a logistic regression.

4.3 Results

To understand which aspects are an improvement over direct data augmentation, an ablation study of the training tasks is done. The benchmark is the conditional augmentation, as proposed by Kumar et al. (2020). We implement our own version to control the experimental setting and obtain results for the new datasets.

Table 4.5: Overview of the steps in the ablation study.

No augmentation
+ Benchmark Augmentation
+ Span-Extractive Fine-Tuning
+ Span-Extractive Pre-Training
+ SpanBERT Training
+ Heterogeneous Augmentation
+ Probabilistic Labels
All Data

First, the model is trained on the small, expert-labeled dataset only. Next, all steps of the methodology are added to observe their impact. First, the pre-training steps are added, starting with span-extractive fine-tuning. Next, making use of weak supervison by including the span-extractive pre-training. Finally, improving the overall span extraction and domain-specific augmentation with SpanBERT. The heterogeneous augmentation expands the benchmark augmentation with the attention-based sampling of the mask positions and error analysis based token selection. Finally, the probabilistic labels are used instead of binary labels. The performance is also compared to a model trained with all data in the training set, which is the reference for maximum achievable result with our implementation of BERT. The extrinsic metrics are chosen to be in line with the GLUE benchmark. Note that for the extrinsic evaluation, when the augmentation model is used as downstream classifier, it has *only* been pre-trained up to the included steps. For example, at the benchmark augmentation step, the only additional training step compared to the base classifier is the augmentation training, no other training, like SpanBERT, is done. This setup is chosen to match how the methodology would be used in practice, given a choice of steps included.

For the augmentation training, the expert-labeled dataset is repeated 10 times, such that different tokens are masked, and trained for 15 epochs with early stopping when the validation *loss* increases. For the heterogeneous augmentation, the expert-labeled dataset is repeated twice, with different masks, wherefore the final combined dataset is three times its original size. This choice is made such that there are more augmented sentences relative to original examples, to make the results more sensitive to the quality of the augmentations. For the extrinsic evaluation, the models are trained with an unbounded number of epochs, but with early stopping until the validation *accuracy* decreases. This strategy prevents that the difference between results may be attributed to the number of training epochs, as every configuration is trained based on the same criteria for optimal performance. Furthermore, it accounts for the smaller dataset in the case of no augmentation. Selecting a maximum token length of a sequence is a trade-off between performance and required computational resources, as attention is calculated with respect to all other tokens, thus increasing exponentially. Therefore, the maximum token length for all tasks is set to 200, which is 4 times the longest mean token length (QNLI), while still allowing for reasonable batch sizes with our resources. The experiments are repeated 15 times with different expert-labeled datasets, sampled from the training data to account for randomness. In the next sections, the results for each task are discussed, as well as general findings from the experiments.

4.3.1 SST-2

The SST-2 task is often used in text-based data augmentation research. It is included both to validate and compare to Kumar et al. (2020). In Table 4.6, the results of the ablation study are given. Our implementation, using all data, has a mean accuracy of 93.4, and Devlin et al. (2018) reported an accuracy of 93.5, thus validating that our methodology works as expected. The proposed methodology achieves a 1.5 absolute point increase over the benchmark, and a 4.2 point increase over not using any augmentation, while reducing variance.

The accuracy without any augmentation - and just 1% of the data - is 83%, which is in line with the expectation that SST-2 is a relatively easy task. However, the standard deviation is almost 4 times larger than for the configurations with augmentation, showing the sensitivity to which observations are sampled from the training data. The benchmark augmentation improves the accuracy with a 2 to 3 point increase, and strongly reduces the standard deviation.

The base model extrinsic evaluation improves slightly when adding the span extraction steps, but improves mostly from SpanBERT. This is also represented in the jump in type-token ratio from 8.9 to 14.1, and improved semantic fidelity from 87.8 to 89.0. The end-to-end augmentation model does improve slightly at every step, but performs less well than the base model up to the addition of SpanBERT. This result may be explained by the augmentation model being trained too much on an easy task with little data, and therefore over-fitting sooner to a local optimum, whereas the base model has not yet been trained on SST-2, and may therefore optimize to a better set of weights. The influence of the pre-trained solution may also be the reason for the augmentation model generally having larger standard deviations than the base model. When extending the augmentation strategy, the augmentation model outperforms the base model, yet - with the increased standard deviation - it is questionable whether this improvement is significant. The inclusion of heterogeneous augmentation and probabilistic labels minimized the standard deviation for the base model, gave the best performance overall for the augmentation model and maximized the semantic fidelity.

Table 4.6: Results from the ablation study for the SST-2 task. All metrics are reported as the mean and standard deviation over the repeated experiments, multiplied by 100. The extrinsic evaluation is measured in accuracy, both for a newly initialized BERT-base model, and the augmentation model (AM). The type-token ratio is the number of unique tokens divided by total number of tokens predicted. The semantic fidelity is the accuracy of the labels of the augmented observations, with respect to predictions made by a model trained on all data available.

Configuration	Accuracy		Type-Token Ratio	Semantic Fidelity
	Base	AM		
No Augmentation	83.3 (7.8)			
Benchmark Aug.	86.0 (2.3)	85.2 (2.4)	9.2 (0.7)	87.3 (1.0)
+ SpEx Fine-Tuning	86.2 (1.5)	85.2 (1.6)	9.0 (0.4)	86.8 (1.2)
+ SpEx Pre-Training	86.4 (1.4)	86.4 (2.1)	8.9 (0.7)	87.8 (1.3)
+ SpanBERT Training	**87.2 (1.3)**	87.1 (1.6)	14.1 (0.7)	89.0 (1.6)
+ Heterogenous Aug.	86.9 (1.5)	87.3 (1.5)	**14.3 (0.7)**	89.0 (1.4)
+ Probabilistic Labels	86.6 (1.1)	**87.5 (1.7)**	14.2 (0.8)	**89.6 (1.3)**
All Data	93.4 (1.4)			

In Figures 4.3 and 4.4, boxplots are shown for the extrinsic results of the base and augmentation models respectively. When using no augmentation, the performance is more sensitive to which training data is sampled for the experiments, as is shown by the large spread. The benchmark augmentation greatly reduces the spread, which is also reflected in the standard deviations in Table 4.6. This is in line with the theoretical proof from Dao et al. (2019), stating that augmentation improves robustness.

Note that for some methods, the median (second quartile) is very close to the first, or, third quartile. For example, in case of the heterogeneous step (yellow or light gray for black and white printing), we found that two thirds of the experiments are very close to 86%, and the remaining third is close to 89%. Thus, the distribution of the performances is not always normal. Around both values the experiments are distributed more normal, but there is an almost binary difference to which group an iteration belongs. The expert-labeled dataset of SST-2 is very small and we theorize these results may indicate whether a certain subclass of problems is contained in the sampled dataset.

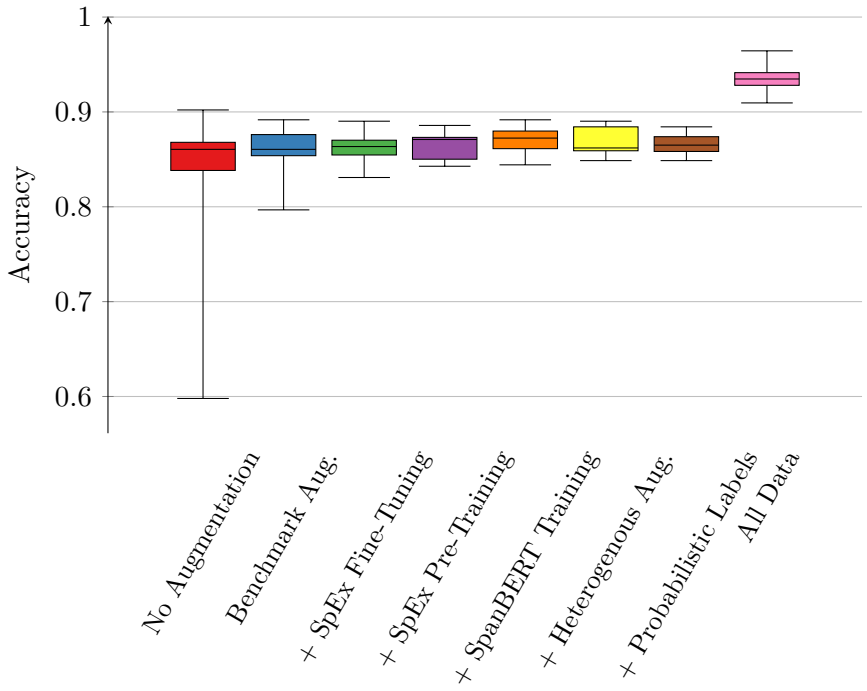

Figure 4.3: Boxplots of the extrinsic results for the SST-2 task, using the BERT-base model as classifier.

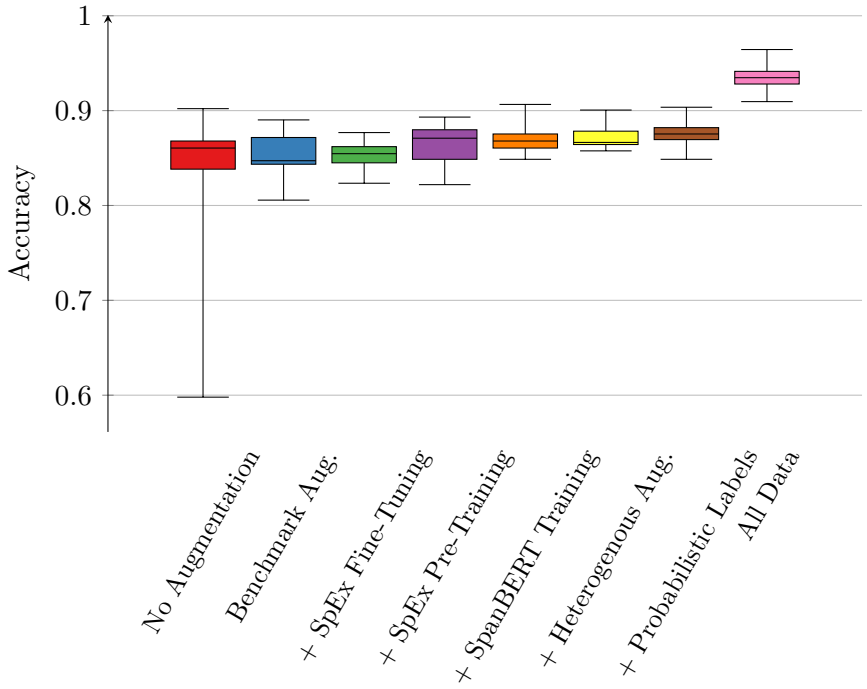

Figure 4.4: Boxplots of the extrinsic results for the SST-2 task, using the augmentation model as classifier.

4.3.2 QQP

The unique property of the QQP task is that it has the largest weakly-labeled dataset, 6 times larger than SST-2 and 3.5 times larger than QNLI (Table 4.3). The extrinsic results are shown in Table 4.8, as the dataset is imbalanced, both accuracy and macro-F1 are reported. The intrinsic results are given in Table 4.7. Again, our results using all data are very close to Devlin et al. (2018). We achieve a 5.8 absolute accuracy point increase over not using augmentation, and a 4.4 increase over the benchmark with the base classifier. In contrast to the SST-2 task, the relative increase from the benchmark to our methodology is larger than the gain from adding the benchmark augmentation. We attribute this benefit to the larger-sized weakly-labeled dataset.

The augmentation model-based results strongly outperform the base model when the span-extractive pre-training step - making use of the large weakly-labeled dataset - is added. However, the type-token ratio and semantic fidelity both decrease. Interestingly, despite the intrinsic decrease, the base model improves extrinsically when the span-extractive pre-training is included, compared to the span-extractive fine-tuning, all be it with a 0.5 instead of a 4.7 point increase, like for the augmentation model. Next, the inclusion of SpanBERT training yields a 2.5 point increase in type-token ratio, as is expected considering it is a masked language modeling task.

The extensions of the augmentation further improve the results. For the base model, the extensions give a 0.7 absolute accuracy point increase compared to SpanBERT, but also increase the standard deviation by 0.2. For the augmentation model, the heterogeneous method only improves the accuracy by 0.2, but also decreases the standard deviation. Finally, when looking at the macro-F1 scores, they confirm the findings from the accuracy-based results.

Table 4.7: Intrinsic results from the ablation study for the QQP task. All metrics are reported as the mean and standard deviation over the repeated experiments, multiplied by 100. The type-token ratio is the number of unique tokens divided by total number of tokens predicted. The semantic fidelity is the accuracy of the labels of the augmented observations, with respect to predictions made by a model trained on all data available.

Configuration	Type-Token Ratio	Semantic Fidelity
Benchmark Aug.	13.4 (1.5)	86.7 (1.6)
+ SpEx Fine-Tuning	13.0 (1.8)	86.3 (1.6)
+ SpEx Pre-Training	11.7 (2.1)	85.9 (1.6)
+ SpanBERT Training	14.2 (0.8)	87.4 (0.8)
+ Heterogenous Aug.	**14.3 (0.9)**	**87.5 (0.8)**
+ Probabilistic Labels	**14.3 (0.9)**	87.3 (0.9)

Table 4.8: Extrinsic results from the ablation study for the QQP dataset. All metrics are reported as the mean and standard deviation over the repeated experiments, multiplied by 100. The extrinsic evaluation is measured in accuracy and macro-F1, both for a newly initialized BERT-base model, and the augmentation model (AM).

Configuration	Accuracy		Macro-F1	
	Base	AM	Base	AM
No Augmentation	75.6 (3.5)		73.5 (8.4)	
Benchmark Aug.	77.0 (1.3)	76.4 (1.3)	76.3 (1.2)	75.4 (1.4)
+ SpEx Fine-Tuning	76.6 (1.0)	76.1 (1.3)	75.8 (1.0)	75.2 (1.2)
+ SpEx Pre-Training	77.1 (1.0)	80.8 (1.5)	76.1 (1.1)	79.8 (1.5)
+ SpanBERT Training	76.9 (1.3)	81.2 (1.4)	76.0 (1.3)	79.9 (1.5)
+ Heterogenous Aug.	77.4 (1.3)	**81.4 (1.2)**	76.5 (1.3)	**80.2 (1.4)**
+ Probabilistic Labels	**77.6 (1.5)**	81.3 (1.2)	**76.6 (1.6)**	80.1 (1.3)
All Data	88.6 (1.5)		87.9 (1.5)	

In Figures 4.5 and 4.6, boxplots are shown for the extrinsic results of the base and augmentation models respectively. Using the newly initialized classifier, there is no large improvement over the benchmark. However, when the augmentation model is used as classifier, and is trained on weakly-labeled data, there is a very clear upward-shift. When adding the additional steps to the procedure, the spreads of the boxplots decrease. Note that for the base model, the pattern where the median can be close to the first or third quartile, is observed again. In the case of the addition of SpanBERT, the median actually improves, but there are also more negative outliers, explaining the mean accuracy reduction of 0.2 compared to span-extractive pre-training in Table 4.8. For the augmentation model classifier, however, the medians are almost perfectly centered. This implies that pre-training on a larger dataset - than for example SST-2 - makes the downstream classifier more robust, showing the benefits of weak supervision at scale.

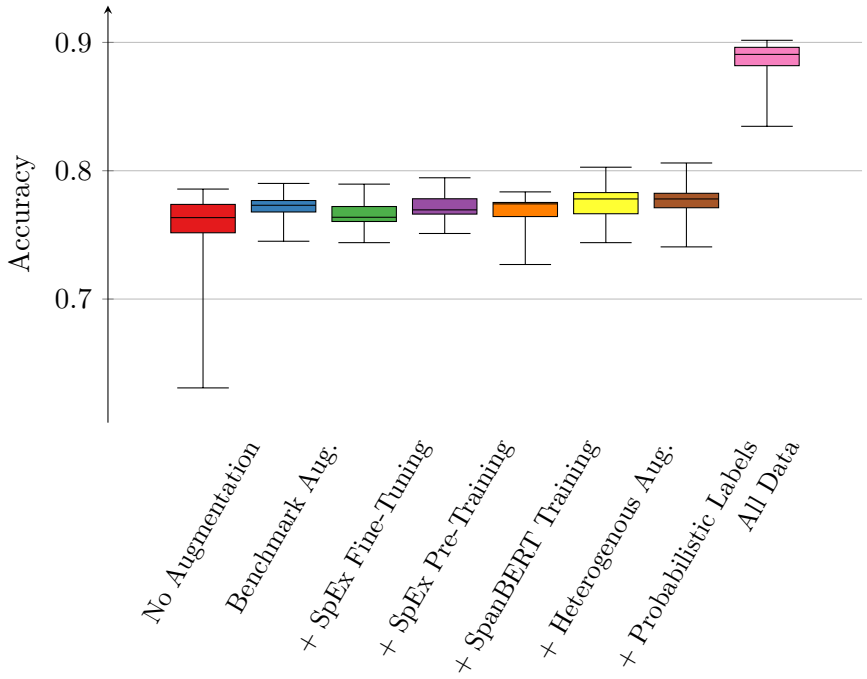

Figure 4.5: Boxplots of the extrinsic results for the QQP task, using the BERT-base model as classifier.

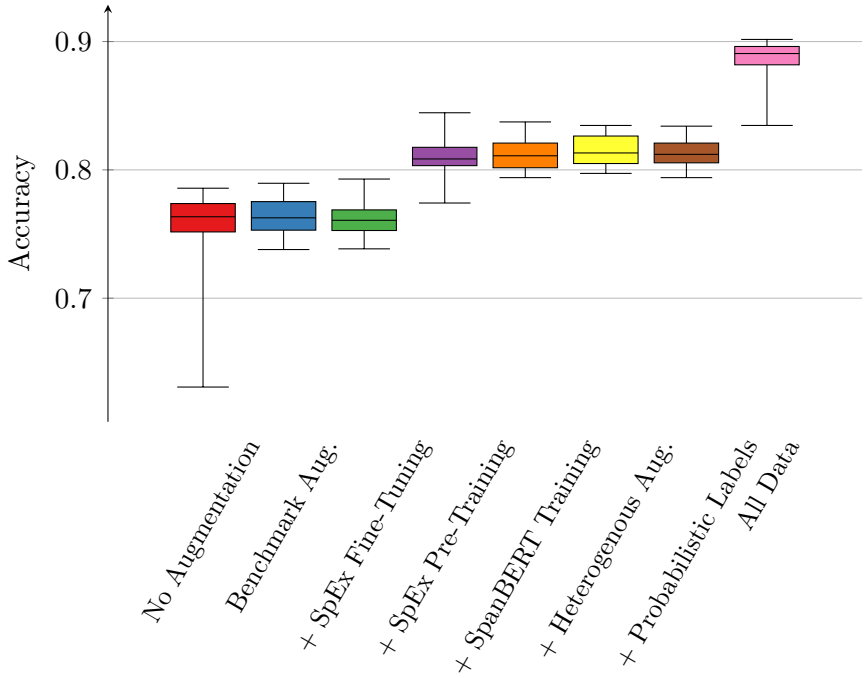

Figure 4.6: Boxplots of the extrinsic results for the QQP task, using the augmentation model as classifier.

4.3.3 QNLI

Recall that the QNLI task has the largest average number of tokens in its observations (Table 4.2). On average, the sampled, expert-labeled dataset has 1,047 observations, which is 772 less than QQP. Therefore, with QNLI, the importance of sequence length can be compared to the influence of the number of sequences (dataset size) for the training of the augmentation model, as both determine the total number of tokens seen. In Table 4.9, the results of the ablation study are given. We achieve a 9.0 absolute point increase over not using augmentation and a 3.0 increase over the benchmark.

For the BERT-base classifier, all but one extensions underperform compared to the benchmark. The exception is the addition of heterogeneous augmentation, with a 0.6 absolute point increase compared to the benchmark, and a 1.2 point increase compared to addition of SpanBERT before it. Recall that in the QNLI task, it needs be determined whether a sequence of text contains the answer to a posed question. It could be that this task is extra sensitive to a few words in a sentence that are key to the classification, and the attention-based sampling improves on selection of which words are augmented. However, that would not explain why the performance decreases again when probabilistic labels are added afterwards. This, therefore is a suggestion for future research.

The augmentation model classifier does outperform the benchmark at every step. The largest improvement comes, like for QQP, from the addition of the weakly-labeled dataset. The extra steps only marginally improve the extrinsic and intrinsic performances. Therefore, we assume that the improvements over the benchmark come from domain-specific pre-training in general, but are not necessarily dependent on which task is used for the pre-training. However, it is worth noting that the 6.0 absolute point increase from the benchmark - compared to not using any augmentation - is very large. We theorize that the benefit from augmentation to the QNLI task, is more in adding noise than actually creating new qualitative samples. For future work, it would be interesting to compare the results to an augmentation method that is not conditional, such as standard masked language modeling.

In Figures 4.7 and 4.8, boxplots are shown for the extrinsic results of the base and augmentation models respectively. For the base model classifier, the SpanBERT boxplot shows a very narrow spread between the first and third quartile, but there is a negative outlier of 70.7% accuracy, when omitting the outlier, the results change from 76.0 (2.0) to 76.4 (1.4). Furthermore, we found that the median of the benchmark is 77.2%, and for the heterogeneous augmentation, the median is

Table 4.9: Results from the ablation study for the QNLI task. All metrics are reported as the mean and standard deviation over the repeated experiments, multiplied by 100. The extrinsic evaluation is measured in accuracy, both for a newly initialized BERT-base model, and the augmentation model (AM). The type-token ratio is the number of unique divided by total number of tokens predicted. The semantic fidelity is the accuracy of the labels of the augmented observations, with respect to predictions made by a model trained on all data available.

Configuration	Accuracy		Type-Token Ratio	Semantic Fidelity
	Base	AM		
No Augmentation	70.6 (11.1)			
Benchmark Aug.	76.6 (1.7)	77.0 (1.1)	13.8 (0.5)	84.8 (0.8)
+ SpEx Fine-Tuning	76.0 (2.1)	77.1 (1.4)	13.1 (0.5)	83.9 (0.6)
+ SpEx Pre-Training	75.9 (1.9)	79.2 (1.4)	12.7 (0.5)	84.1 (1.2)
+ SpanBERT Training	76.0 (2.0)	79.5 (1.1)	**15.6 (0.4)**	85.5 (1.0)
+ Heterogenous Aug.	**77.2 (1.4)**	**79.6 (1.3)**	15.5 (0.3)	**85.8 (1.0)**
+ Probabilistic Labels	76.3 (1.5)	79.6 (1.5)	15.5 (0.4)	85.6 (0.9)
All Data	88.7 (1.0)			

76.9%. It appears that the heterogeneous augmentation outperforms the benchmark in Table 4.9, because of better robustness, having less outliers. This is also visible in the boxplot when looking at the spreads in comparison to the median lines.

The augmentation model classifier, in comparison to the base model, has smaller and more consistent spreads. This robustness is also reflected in the standard deviations in Table 4.9. Again, a strong upward-shift occurs when making use of weak supervision. Interestingly, the extensions following mainly seem to improve due to positive outliers, whereas the median values are very close.

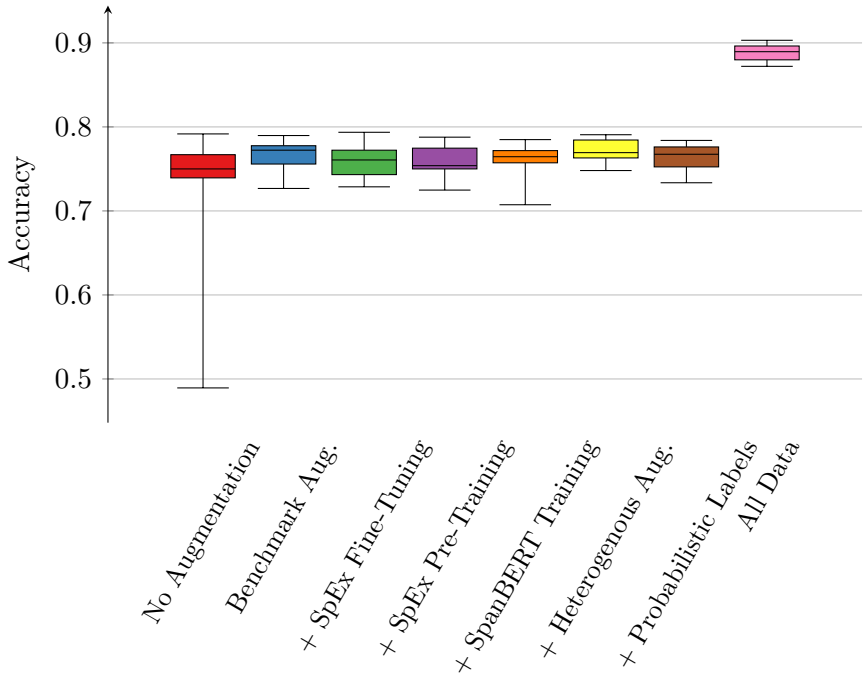

Figure 4.7: Boxplots of the extrinsic results for the QNLI task, using the BERT-base model as classifier.

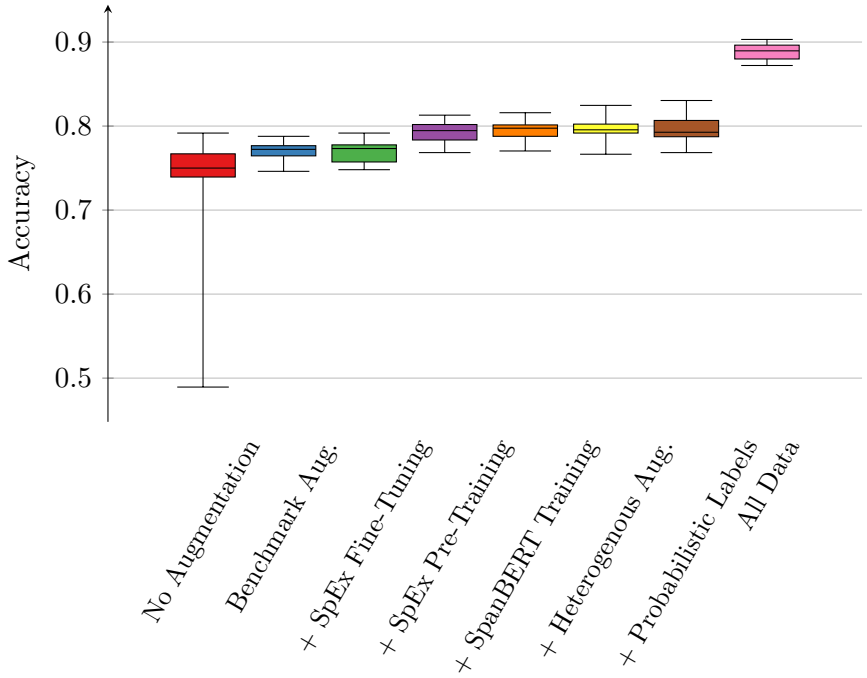

Figure 4.8: Boxplots of the extrinsic results for the QNLI task, using the augmentation model as classifier.

4.3.4 General Findings

For all three tasks, the best-performing configuration is the proposed methodology, sometimes excluding the probabilistic labels, and using the augmentation model as final classifier. The benefit from weak supervision and transfer learning is proportional to the amount of unlabeled data available. The heterogeneous augmentation and probabilistic labels mostly improved the robustness of the downstream classifier. Not using any augmentation, for all tasks, results in a wide spread in the extrinsic accuracies across experiments, showing the necessity for robustness.

SST-2 is the only task shared with the other research in the field. Data augmentation is mostly tested on topic classification or sentiment analysis. To the best of our knowledge, this is the first paper to apply textual augmentation to any natural language understanding task. One could argue that, intuitively, a topic classification task is easier to augment. However, to our surprise, both the QQP and QNLI tasks have greater absolute performance improvements than SST-2. This might be related to the spread in performance between using the small sampled dataset, and when all data is available, or simply because QQP and QNLI have more data. When comparing the relative performance improvements, SST-2 still has the smallest gain, but the results are closer. The sampled dataset for SST-2 has the smallest number of observations, but the baseline without augmentation is 83%, compared to 76% for QQP and 71% for QNLI. Thus, SST-2 is clearly an easier task for a BERT classifier. Therefore, even though SST-2 intuitively is more suited for augmentation, there is less performance to be gained from it, similarly to how a logistic regression will be closer to a BERT model in performance for a simple task than for a complex task.

For QNLI, both the benchmark and best type-token ratios are larger than for either the SST-2 or QQP tasks. QQP has more unlabeled data, but a smaller average number of tokens in the sequences (Table 4.2). We hypothesize that the better type-token ratio is explained by the mean token length. Recall that, in our implementation, SpanBERT uses span lengths drawn from a geometric distribution, with as mean, 15% of the number of tokens of that specific observation. Therefore, the span lengths in QNLI are larger on average (7.5 tokens) than the spans in QQP (4.6 tokens). This would also explain the smaller type-token ratio for SST-2, where the average span length is only 2.0 tokens. However, the difference might also be explained simply by the difference in corpora, as well as possible differences in similarities to the datasets used by Devlin et al. (2018) for the initial training.

5 Discussion and Future Work

During preliminary experiments, we found that setting the correct hyperparameters is essential to getting satisfactory outcomes. However, the ablation study is computationally expensive. For example, a single iteration for QQP, on an NVIDIA V100 GPU with 16GB of RAM, takes over 12 hours. For the experiments, we used up to 8 machines, but remained constraint in the number of configurations that could feasibly be compared. Therefore, there are numerous variations on our experiments that could be done to further understand the methodology. These variations include:

- Different levels of simulated weak supervision noise, including using the original expert labels without any noise (the original training data);
- Which natural words are used as textual labels, both in terms of length and meaning;
- Different sizes of sampled expert-labeled dataset;
- How often the expert-labeled dataset is repeated, with different masks for both augmentation training and heterogeneous augmentation;
- Masking more than 15% of a sequence.

By including an early stopping rule, most steps in the methodology are trained for an optimized number of epochs. However, the MLM tasks - both SpanBERT training and the augmentation training - never reached the early stopping point where the validation loss decreases, thus, always trained for the maximum amount of epochs. Recall that, during augmentation training, the dataset is repeated 10 times with different masks, and trained for, at most, 15 epochs. When training for 5 epochs only, the augmentation model would mainly predict periods and commas. Only once we increased to 10 epochs, the augmentations empirically started to become semantically valid. We also found that, from time-to-time, the augmentations were suspiciously precise. For example, when predicting the name of a specific person or movie. On further inspection, we found that SST-2 reuses sub-samples of the same movie review as individual observations, QQP pairs questions multiple times, and QNLI pairs multiple answer sequences to the same questions. However, as masked language modeling does not require manual labels, one could theoretically even train on the test set. We do not consider the duplicates a problem for the augmentation training directly, as both the augmentation training and inference are done on the expert-labeled data regardless. The duplicates might, however, interfere with the early stopping rule, if the validation set contains

duplicates, as over-fitting on the training data would not necessarily decrease the validation loss. In future work, it would be interesting to determine the upper bound for the number of epochs for the augmentation training.

BERT is trained to process a sequence that contains either one segment, like a movie review, or two segments, like two questions or a question and an answer, separated by the "[SEP]" token. The span-extractive classification, proposed by Keskar et al. (2019), appends the natural classification to the end of the first segment, if there are two segments used, and to its own segment, if available. We diverged from this formulation by always having the textual labels at the start of the sequence. This choice was made, such that the text structure of the span-extractive classification is similar to the structure of the conditional augmentation, as proposed by Kumar et al. (2020). Furthermore, we randomly shuffle the order of the labels to prevent the classifier from training on the positions rather than the tokens. In future work, our formulation could be compared to the original span-extractive formulation to validate the design choices made.

During the development of the methodology, we considered multiple ways to calculate probabilistic labels based on the augmentation. The proposed version, the average probability (Equation 3.5.2), could be improved in future research. Initially, the idea was to use the probability of how valid a token is. However, in masked language modeling, the probabilities are not independent scores, relative to other tokens in the vocabulary. Therefore, even the most likely token typically only has a probability of 5%. Thus, in practice, the probabilistic label is roughly equal to the percentage of non-masked tokens in a sequence, and less influenced by the validity of the selected tokens. Furthermore, the intrinsic evaluation of the semantic fidelity uses accuracy, as proposed by Kumar et al. (2020), but preferably a metric, such as cross-entropy, is used, that incorporates the probabilistic label.

Finally, the weak supervision is simulated with a random but homogeneous process. When using a method like Snorkel, the premise is that, depending on the heuristics, some labels are of higher confidence than others. While the different confidence levels are simulated, the underlying process is not. It might be that a certain subclass of observations is covered by multiple heuristics, whereas other observations are not covered at all. BERT might generalize the heuristics for specific subsets, while not improving for the other subsets. Most likely, the implications are dependent on both the domain and quality of the heuristics. It would be of great interest to validate our methodology with a real-life weak supervision method, although - due to the dependence on specific heuristics and domains - the outcome would be hard to generalize.

6 Conclusion

In this work, we propose a new methodology for data augmentation, using weak supervision and span extraction. Multiple methods of transfer learning and pre-training are combined that were previously considered disjoint solutions to the same problem. We outperform the benchmark for the SST-2 task by 1.5, QQP task by 4.4, and QNLI task by 3.0 absolute accuracy points. This shows that, when a weak supervision method is available, it is beneficial to combine it with data augmentation. Data augmentation not only improves the average performance, but also is more robust when working with small datasets. Additionally, the downstream model improves further when it has been pre-trained using the proposed methodology. This implies that it is beneficial to use the model used for the augmentation, as downstream model as well, fine-tuning it on the observations it has created itself. Furthermore, we show that data augmentation is not only possible for natural language understanding, but even more effective than for the typically used, simpler task.

In a business setting, expert labels are often expensive to obtain, and labeling definitions might be subject to change. For example, in medical applications, specialized doctors need to label scans, and new findings might invalidate previous practices. By investing in labeling heuristics, experts can incorporate knowledge for an unlabeled dataset of unconstrained size, and a much smaller expert-labeled dataset is needed for fine-tuning and validation only. Thus, when manually labeling enough data for a neural network is economically infeasible, investing in weak supervision is a good alternative. We argue that, when weak supervision is used for natural language processing, it should be combined with our methodology. Depending on the application, a point increase in performance can have a large impact economically. Aside from increasing overall performance, our method improves the robustness, which is perhaps even more valuable in a production setting. Our methodology does not require any domain-specific adjustments, and thus, implementation is only a trade-off between improved performance and computational cost, and will often be a positive return on investment.

To conclude, weak supervision's general increase in performance, as a weakly-labeled dataset grows, also holds when applied to data augmentation. Therefore, in an era where unlabeled data is abundant, computational resources are cheap and Moore's law is still valid, combining weak supervision and data augmentation is a scalable and effective way to improve downstream models.

References

Alammar, J. (2018). The illustrated transformer. http://jalammar.github.io/illustrated-transformer/. (accessed: 2020-05-4).

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Dao, T., Gu, A., Ratner, A., Smith, V., De Sa, C., and Ré, C. (2019). A kernel theory of modern data augmentation. In *International Conference on Machine Learning (ICML 2019)*, pages 1528–1537. PMLR.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pages 248–255. IEEE.

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

Guo, H., Mao, Y., and Zhang, R. (2019). Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Hancock, B., McCreery, C., Chami, I., Chen, V. S., Wu, S., Dunnmon, J., Varma, P., Lam, M., and Ré, C. (2019). Massive multi-task learning with Snorkel MeTaL: Bringing more supervision to bear. `https://dawn.cs.stanford.edu/2019/03/22/glue/`. (accessed: 2020-06-07).

Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural computation*, 9(8):1735–1780.

Iyer, S., Dandekar, N., and Csernai, K. (2017). First quora dataset release: Question pairs. `https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`.

Jin, D., Jin, Z., Zhou, J. T., and Szolovits, P. (2020). Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *2020 AAAI Conference on Artificial Intelligence (AAAI 2020)*, volume 34, pages 8018–8025.

Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Keskar, N. S., McCann, B., Xiong, C., and Socher, R. (2019). Unifying question answering, text classification, and regression via span extraction. *arXiv preprint arXiv:1904.09286*.

Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Kumar, V., Choudhary, A., and Cho, E. (2020). Data augmentation using pre-trained transformer models. In *2nd Workshop on Life-long Learning for Spoken Language Systems (LifeLongNLP 2020)*, pages 18–26. ACL.

Lim, B., Arik, S. O., Loeff, N., and Pfister, T. (2019). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *arXiv preprint arXiv:1912.09363*.

Liu, X., He, P., Chen, W., and Gao, J. (2019a). Multi-task deep neural networks for natural language understanding. In *57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 4487–4496. ACL.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Loshchilov, I. and Hutter, F. (2018). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR 2019)*. OpenReview.net.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *26th International Conference on Neural Information Processing Systems (NIPS 2013)*, pages 3111–3119. Curran Associates, Inc.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Natarajan, N., Dhillon, I. S., Ravikumar, P., and Tewari, A. (2013). Learning with noisy labels. In *26th International Conference on Neural Information Processing Systems (NIPS 2013)*, pages 1196–1204. Curran Associates, Inc.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., and Tran, D. (2018). Image transformer. In *International Conference on Machine Learning (ICML 2018)*, pages 4055–4064. PMLR.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global vectors for word representation. In *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543. ACL.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2018)*, pages 2227–2237. ACL.

Phang, J., Févry, T., and Bowman, S. R. (2018). Sentence Encoders on STILTs: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Quteineh, H., Samothrakis, S., and Sutcliffe, R. (2020). Textual data augmentation for efficient active learning on tiny datasets. In *2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 7400–7410. ACL.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. *OpenAI Blog*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. In *2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2383–2392. ACL.

Ratner, A., Bach, S., Ehrenberg, H., Fries, J., Wu, S., and Ré, C. (2020). Snorkel: rapid training data creation with weak supervision. *The VLDB Journal*, 29:709–730.

Ratner, A., Hancock, B., Dunnmon, J., Goldman, R., and Ré, C. (2018). Snorkel MeTaL: Weak supervision for multi-task learning. In *Second Workshop on Data Management for End-To-End Machine Learning (DEEM 2018)*, pages 1–4. ACM.

Ratner, A., Sa, C. D., Wu, S., Selsam, D., and Ré, C. (2016). Data programming: creating large training sets, quickly. In *30th International Conference on Neural Information Processing Systems (NIPS 2016)*, pages 3574–3582. Curran Associates, Inc.

Roemmele, M., Gordon, A. S., and Swanson, R. (2017). Evaluating story generation systems using automated linguistic analyses. In *2017 Workshop on Machine Learning for Creativity (ML4Creativity 2017)*, pages 13–17. ACM.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, pages 5149–5152. IEEE.

Simard, P. Y., LeCun, Y. A., Denker, J. S., and Victorri, B. (1998). Transformation invariance in pattern recognition—tangent distance and tangent propagation. In *Neural Networks: Tricks of the Trade*, pages 239–274. Springer.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 1631–1642. ACL.

Taylor, W. L. (1953). Cloze Procedure: A new tool for measuring readability. *Journalism and Mass Communication Quarterly*, 30(4):415.

Varma, P. and Ré, C. (2018). Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment*, volume 12, pages 223–226.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In *31st International Conference on Neural Information Processing Systems (NIPS 2017)*, page 6000–6010. Curran Associates Inc.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019). SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *33rd Annual Conference on Neural Information Processing Systems (NIPS 2019)*. Curran Associates, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP 2018)*, pages 353–355. ACL.

Wei, J. and Zou, K. (2019). EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pages 6382–6388. ACL.

Wu, X., Lv, S., Zang, L., Han, J., and Hu, S. (2019). Conditional BERT contextual augmentation. In *19th International Conference on Computational Science (ICCS 2019)*, pages 84–95. Springer.

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., and Hsieh, C.-J. (2020). Large batch optimization for deep learning: Training BERT in 76 minutes. In *8th International Conference on Learning Representations (ICLR 2020)*. OpenReview.net.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations (ICLR 2018)*. OpenReview.net.

Appendix

A List of Symbols

This section serves as a reference for all symbols used in the methodology.

Table A.1: Overview of all symbols used in Section 3.

Symbol	Description
X	Sequence of tokens
x_i	Token in sequence X
$\mathbf{x}_i$	Final BERT embedding of token x_i
$\hat{x}_{\pi_k}$	Selected replacement token for m_k
N	Number of tokens in sequence X
Π	Set of masked token indices in sequence X
π_k	Masked token index
K	Number of masked tokens in sequence X
M	Set of masked tokens in sequence X
m_k	Masked token for x_{π_k}
$\mathbf{m}_k$	Final BERT embedding of token m_k
$\mathbf{m}'_k$	Adjusted BERT embedding of token m_k for Span Boundary Objective
V	Number of tokens in vocabulary
$\mathbf{p}_k$	Positional embedding for BERT input
$\mathbf{w}_v$	Weight vector in activation function
b_k	Bias in activation function
l	Geometrically distributed variable for span length
s, e	Start and end tokens for span masking
$\mathcal{L}$	Loss function
$\mathbf{s}$	Token selector vector for span extraction
y	True label
$\tilde{y}$	Probabilistic label from weak supervision
y^*	Probabilistic label from augmentation
e	Target analysis error, used as complexity measure
LB, UB	Task-wide lower and upper bound probabilities for token replacement candidates
LB_s	Sample specific lower bound for token replacement candidates
$\mathbf{h}_j$	Hidden state embedding
H	Hidden size of embedding
L	Number of transformer blocks
A	Number of attentions head for multi-headed attention

B List of Acronyms

This section serves as a reference for all acronyms used throughout the paper.

Table B.1: Overview of all acronyms used, in alphabetical order.

Acronym	Description
AM	Augmentation model
BERT	Bidirectional Encoder Representations from Transformers (Devlin et al., 2018)
CBERT	Conditional BERT (Wu et al., 2019)
CLS	Classification
DA	Data augmentation
DAWSON	Data Augmentation using Weak Supervision On Natural Language
EDA	Easy Data Augmentation (Wei and Zou, 2019)
ELMo	Embeddings from Language Models (Peters et al., 2018)
GloVe	Global Vectors (Pennington et al., 2014)
GLUE	General Language Understanding Evaluation benchmark (Wang et al., 2018)
GPT	Generative Pre-trained Transformer (Radford et al., 2018, 2019; Brown et al., 2020)
GPU	Graphics processing unit
LAMB	Layer-wise Adaptive Moments (You et al., 2020)
LB	Lower bound
LSTM	Long Short Term Memory (Hochreiter and Schmidhuber, 1997)
MCC	Matthews correlation coefficient (Matthews, 1975)
ML	Machine learning
MLM	Masked language modeling
MT-DNN	Multi-Task Deep Neural Networks (Liu et al., 2019a)
NLP	Natural language processing
NLU	Natural Language Understanding
NSP	Next sentence prediction
PAD	Padding
POS	Part of speech
QNLI	Question-answering natural language inference (Rajpurkar et al., 2016)
QQP	Quora Question Pairs (Iyer et al., 2017)
RNN	Recurrent neural network
RoBERTa	Robustly optimized BERT approach (Liu et al., 2019b)
RTE	Recognizing Textual Entailment
SBO	Span Boundary Objective (Joshi et al., 2020)
SEP	Separator
SpEx	Span-Extractive
SQuAD	Stanford Queston Answering Dataset (Rajpurkar et al., 2016)
SST	Stanford Sentiment Treebank (Socher et al., 2013)
STILTs	Sentence Encoders on Supplementary Training on Intermediate Labeled-data Tasks (Phang et al., 2018)
UB	Upper bound
WS	Weak supervision