The New England Journal of Statistics in Data Science Volume 0, 1–15 (2022)
DOI: https://doi.org/10.51387/22-NEJSDS8

Conformal Prediction for Text Infilling and Part-of-Speech Prediction

Neil DEY, Jing DING, Jack FERRELL, Carolina KAPPER, Maxwell LOVIG, Emiliano PLANCHON, and Jonathan P. WILLIAMS*

Abstract

Modern machine learning algorithms are capable of providing remarkably accurate point-predictions; however, questions remain about their statistical reliability. Unlike conventional machine learning methods, conformal prediction algorithms return confidence sets (i.e., set-valued predictions) that correspond to a given significance level. Moreover, these confidence sets are valid in the sense that they guarantee finite sample control over type 1 error probabilities, allowing the practitioner to choose an acceptable error rate. In our paper, we propose inductive conformal prediction (ICP) algorithms for the tasks of text infilling and part-of-speech (POS) prediction for natural language data. We construct new ICP-enhanced algorithms for POS tagging based on BERT (bidirectional encoder representations from transformers) and BiLSTM (bidirectional long short-term memory) models. For text infilling, we design a new ICP-enhanced BERT algorithm. We analyze the performance of the algorithms in simulations using the Brown Corpus, which contains over 57,000 sentences. Our results demonstrate that the ICP algorithms are able to produce valid set-valued predictions that are small enough to be applicable in real-world applications. We also provide a real data example for how our proposed set-valued predictions can improve machine generated audio transcriptions.

keywords and phrases: BERT, BiLSTM, Natural language processing, Set-valued prediction, Uncertainty quantification.

1. INTRODUCTION

In recent years, machine learning algorithms have dominated the realm of natural language processing (NLP). Over time, these algorithms have achieved higher and higher accuracy in various NLP tasks. However, such algorithms are specialized for point prediction, and as such, a significant limitation of many machine learning algorithms is that they do not offer any uncertainty quantification to measure how reliable these point predictions are actually correct.

To address this limitation, we make the following contributions. We construct three new conformal prediction-enhanced algorithms for two important NLP tasks. The algorithms we construct inherently provide uncertainty quantification guarantees by yielding calibrated set-valued predictions at any user-specified type 1 error rate. In particular, we apply conformal prediction to the masked language modeling (MLM) and POS tagging tasks; to our knowledge, conformal prediction has not yet been applied to these two key tasks in NLP. We construct new conformal prediction-enhanced BERT and BiLSTM algorithms for POS tagging and a new conformal prediction-enhanced BERT algorithm for MLM. Using the Brown Corpus [14], we empirically demonstrate that BERT provides smaller prediction sets for POS tagging than a BiLSTM model, and we show that BERT generates usefully small prediction sets for MLM. Moreover, we show that these conformal prediction sets achieve their nominal coverage for any level of significance and produce relatively small prediction sets at reasonably high confidence levels. Finally, we provide a real data example to illustrate how our proposed set-valued predictions are effective at improving machine generated audio transcriptions.

Conformal prediction is an approach introduced in [54] that allows, for example, a point prediction method to be extended to form confidence sets, guaranteeing that the set contains the true unknown predictor value with some nominal coverage probability. It has been shown that deep learning architectures such as multilayer perceptrons (MLP), convolutional neural networks (CNN), and gated recurrent units (GRU) often improve in their robustness when enhanced by a conformal prediction algorithm [34]. Conformal prediction has been applied to text classification NLP tasks. For example, similar results are demonstrated in [32] and [31] for conformal prediction-enhanced BERT and artificial neural network (ANN)-based sentiment classification and multi-label text classification, respectively. Other experiments in the literature, such as [38] with deep neural network (DNN)-based multi-label text classifiers and [4] with tree-based classifiers, replicate these

*Corresponding author.

findings for other multi-label classification models. Conformal prediction-enhanced BERT-based models for paraphrase detection are constructed in [16], and a definition and analysis of *credibility* – relevant to Section 4.1 – is provided. Conformal prediction has also been successful in relation classification, identifying relationships between two entities in a sentence, as demonstrated in [11] and for open-domain question answering and information retrieval for fact verification in [12]. To our knowledge, however, conformal prediction has not yet been applied to two key tasks in NLP: the text infilling task and POS tagging.

The text infilling task (also known as the Cloze task) is a standard NLP task, asking a model to "fill in the blank" given an otherwise complete sentence. Since its conception, the task has greatly expanded in scope due to the great success of various text infilling algorithms developed. For example, generative adversarial networks are used to great effect in the MaskGAN algorithm in [10] to generalize the problem to full text generation. Another generalization of the text infilling task was introduced in [36] in the form of the story cloze test, determining the "right ending" to a story. The story cloze test has been further explored in the form of neural network solutions [51] and generative pre-training of language models [46], among other methods. Yet another extension to the text infilling task comes in the form of filling in blanks of arbitrary length, as explored in [61] (utilizing self-attention mechanisms) and in [50] (using the blank language model). Although many techniques have been proposed to solve the text infilling task, such as gradient-search-based inference [28] and infilling by language modeling [8], text infilling in practice has been dominated by the BERT algorithm [7], which uses an MLM pre-training objective to attain word embeddings. Though trained on the text infilling task, the resulting word embeddings remain competitive in many standard NLP tasks.

The POS tagging task is another standard NLP task in which a model assigns the correct grammatical POS to each word in a sentence. This task is unusual in the NLP realm in that the most naive algorithm of simply assigning each word its most common POS already achieves a very high baseline accuracy of roughly 92% [23, Chapter 8, end of Section 2]. The introduction of some classical models such as hidden Markov models (HMM) [25] and conditional random fields (CRF) [26] improved the accuracy to about 96%; more modern techniques currently used such as the BiLSTM proposed in [57] and transformer models such as BERT [7] offer further marginal improvements, reaching about 97–98% accuracy. Similar to the text infilling task, it does not appear that the application of conformal prediction to POS tagging is present in the literature. However, a method of set-valued prediction introduced in [35] has been applied to POS tagging of a middle-lower German corpus in [18], demonstrating more robust predictions than standard POS tagging algorithms, but these set-valued predictions do not offer the guaranteed control over type 1 error probabilities that are inherent in conformal prediction sets. As discussed in [18], POS tagging of historical corpora remains one area where linguistics experts do not necessarily know or agree on the POS for particular words because the languages are no longer in use. In these applications, set-valued predictions are most sensible.

Furthermore, in machine learning applications, since the accuracy of POS tagging is typically high, it can be expected that many set-valued POS predictions will be of size 1, and greater than 1 for occasional ambiguous cases. Accordingly, the set-valued POS tagging algorithms that we contribute combine the speed of automated tagging with the accuracy of manual tagging.

A brief overview of BiLSTM models, transformers and BERT, and conformal predictions is given in Section 2. Section 3 presents our proposed algorithms, followed by a discussion of our empirical studies in Section 4. The utility of the enhanced BERT model for MLM in a realistic setting is illustrated in Section 5 by running the model on missing words from a transcript of a TED Talk generated by automatic speech recognition software, and the paper closes with concluding remarks provided in Section 6. The code and workflow for reproducing our results, along with documented software for implementing our algorithms on new data sets, are available at https://github.com/jackferrellncsu/drums-nlp-codesnapshot.

2. EXISTING MACHINE LEARNING APPROACHES

Currently, the state-of-art methods for MLM tasks are BERT-based [6]. Other models include TagLM [43] and ELMo [44]. TagLM and ELMo both use recurrent neural networks (RNN), and ELMo specifically constructs a two-layer BiLSTM, commonly used as a pre-trained model for the embedding layer for other models. Alternatively, BERT models use transformers instead of an LSTM in the deep embedding layer.

POS tagging takes a sequence of words and assigns each word a particular POS. It is a sequence labeling task because each word can represent a different POS depending on its context. POS tagging is useful in syntactic parsing, reordering in translation, sentiment tasks, text-to-speech tasks, etc. Classic POS labeling algorithms include HMM and linear chain CRF. HMM is a probabilistic sequence model that computes a probability distribution over possible sequences of labels and chooses the label sequence with highest likelihood. However, as a generative model, HMM does not incorporate arbitrary features for unknown words in a clean way. An HMM is implemented in [3] that handles unknown words using suffix features and attains an accuracy of 96.46% on a particular corpus. CRF is a log-linear model that assigns a probability to an entire output (label) sequence with respect to all the possible sequences, given the entire sequence of input words. A CRF method for structure regularization, proposed in [52], achieved 97.36% accuracy on the corpus they

consider (though, this accuracy cannot be directly compared to that reported in [3] due to the difference in data sets).

Modern POS labeling algorithms include RNNs and transformer networks, which both manage to deal directly with the sequential nature of language surrounding a target word. RNN architectures contain a cycle within the network connections, where the value of a unit is directly or indirectly dependent on the earlier output as an input. The BiLSTM architecture has achieved wide attention due to its effectiveness for sequence classification. It solves the "vanishing gradient" problem by forgetting information that is no longer needed, carrying information that is required for decisions to come, and combining the forward and backward network results. Researchers have applied BiLSTMs and obtained accuracies ranging from 97.22% to 97.76% [27, 45, 59, 2, 58, 30]. As an alternative solution, transformers are made up of blocks including self-attention layers, feedforward networks, and custom connections. Transformer based models, such as BERT, are pre-trained on large context corpora and are well-suited for POS tagging.

Although it appears promising that the accuracy of POS tagging has reached 97% for English language texts, the baseline accuracy is 92% [23, Chapter 8, end of Section 2] because many words have only a single POS, and those that have multiple POS overwhelmingly occur with their most common class. However, a single bad tagging in a sentence can lead to a huge error in downstream tasks such as dependency parsing. It is thus more meaningful to view the accuracy of the whole-sentence POS tagging, which is around 55–57% [33]. Researchers have been trying to improve the accuracy of POS tagging via improvements in features, parameters, and learning methods without breakthrough success. Meanwhile, there are concerns regarding the correctness of the treebank and whether POS labels are well-defined to allow us to assign each word a single symbolic label [33]. That is to say, it is possible that the error in POS labeling is due to linguistically justified definitions and cannot be further improved without improvement in the field of linguistics.

One way to deal with the current error in POS tagging is to add associated confidence values for each prediction. All the aforementioned approaches only output a simple point prediction without evaluating how likely it is for each prediction to be correct. The likelihood of each prediction enables us to assess to what extent a prediction can be relied on, and generates alternative POS tags. This serves as a filtering mechanism with regard to the corresponding confidence level and can help avoid the problem that a single mistake in a sentence limits the usefulness of a tagger for downstream tasks. Conformal prediction [48] is well-suited to provide such confidence information on top of the traditional algorithms, and the more computationally feasible, ICP approach for neural network predictions is introduced in [39]. ICP is applied to a binary text classification problem in [32] using a BERT model for contextualized word embeddings. The results show that the prediction accuracy for the BERT classifier was maintained, while the prediction sets calculated using the conformal prediction algorithm provided more useful information. The conformal prediction correctness criterion is expanded in [12] by adding admissible labels to reduce the size of prediction sets, and by filtering out implausible labels early on by using conformal prediction cascades to decrease the computational cost. The application of conformal prediction for "multi-label" text classification using DNNs based on contextualized and non-contextualized word embeddings is considered in [31]. They reduced the computational complexity by eliminating label-sets that would surely have p-values below the specified significance level. Their results show that the context-based classifier with conformal predictions has good performance and small prediction sets that are practically useful. Further work is provided in [13] to expand the use of conformal predictions for information retrieval with a cascading approach, filtering out incorrect options at every step with the hopes of keeping at least one "admissible" option after all the layers. This approach was found to improve both computational and predictive efficiency by giving the model fewer items to sort through at each step.

2.1 Long Short-Term Memory Neural Net

The use of RNNs in NLP tasks is very common due to the sequential nature of language. Unlike feed-forward networks, RNNs are able to take into account all of the preceding words in a variable length sequence with fixed-size input and embedding vectors when making predictions [9]. In language tasks like next word prediction, this is desirable because the more structured the context that a model is learning from, the more accurate the prediction is likely to be.

In machine learning, the goal of a gradient descent algorithm is to minimize the cost function by finding and updating the parameters of the model. With RNNs, using gradient descent with an error criterion for tasks involving long-term dependencies is inadequate and may result in exploding or vanishing gradients [1]. This problem arises when the network updates the weights while back-propagating through time during training [20]. An extremely large gradient will make the model that is being trained unstable, and an extremely small (≈ 0) gradient will make it impossible for the model to learn correlations between events with a high temporal span of dependencies [41]. Moreover, gradient descent becomes less efficient the further apart the inputs are, suggesting that RNNs are not desirable for tasks that require long-term "memory." There have been many theorized solutions to these issues; however, none are as prevalent as gated neural networks [22].

A popular type of gated neural network is the LSTM [21]. LSTMs help prevent vanishing and exploding gradients through the use of a memory cell, which is regulated by the forget (f_t), input (i_t), and output (o_t) gates (see Figure 1). Each of these gates contain a sigmoid activation alongside a

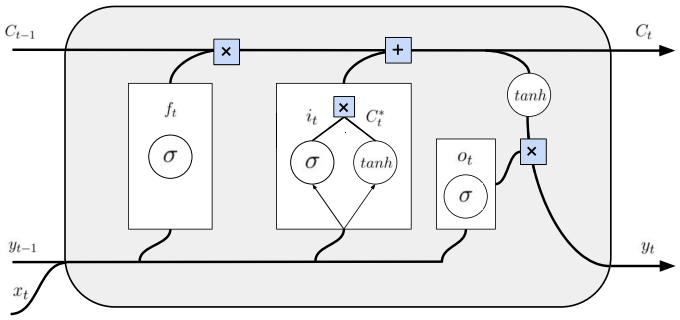

Figure 1: LSTM Memory Cell.

component-wise multiplication operation. The sigmoid layer outputs values that are between 0 and 1 which serve as indicators for the proportion of each component that will be "let through" the gate. The standard reference for describing the architecture and intuition for memory cells is given in [37]. For convenience, we summarize the main ideas in the remainder of this section.

The forget gate (f_t) considers y_{t-1} and x_t, where y_{t-1} is the network output layer at time $t-1$ and x_t is the input vector at time $t \in \mathbb{N}$. These quantities are passed through the vectorized sigmoid function

$$f_t = \sigma([x_t^\top, y_{t-1}^\top] \cdot W_f + b_f),$$

where W_f and b_f are a weight matrix and bias vector, respectively. After passing $[x_t^\top, y_{t-1}^\top]$ through the forget gate, the past cell state C_{t-1} is multiplied component-wise with f_t. Next, as shown in Figure 1, a tanh activation function also evaluated at $[x_t^\top, y_{t-1}^\top]$, but with a different weight matrix W_C and bias vector b_C, is used to create a vector of values in $[-1, 1]$:

$$C_t^* = \tanh\big([x_t^\top, y_{t-1}^\top] \cdot W_C + b_C\big).$$

The input gate is similarly constructed as

$$i_t = \sigma([x_t^\top, y_{t-1}^\top] \cdot W_i + b_i)$$

for weight and bias terms W_i and b_i and the cell is updated as

$$C_t = f_t \odot C_{t-1} + i_t \odot C_t^*,$$

where $\odot$ denotes component-wise multiplication. The $f_t \odot C_{t-1}$ term controls how much of the past cell memory to carry forward, and the $i_t \odot C_t^*$ term controls how much of the updated cell memory to add [17]. Lastly, the cell updates the state y_t as

$$o_t = \sigma([x_t^\top, y_{t-1}^\top] \cdot W_o + b_o)$$
$$y_t = o_t \odot \tanh(C_t),$$

using a final set of weight and bias terms, W_o and b_o.

The implementation of a memory cell like the one above is quite common; however, there is much variety when it comes to the exact details [37]. Examples include GRUs [5], peephole connections [15], and clockwork RNNs [24], among others. The sophisticated nature of these memory cells have proven to work efficiently on NLP problems [49], which is why we consider it a favorable method to combine with a conformal predictor.

2.2 Transformers and BERT Embeddings

Recurrent models, while useful for encapsulating information about the structure of sentences, are extremely computationally expensive in practice. Namely, the sequential nature of such models makes training them impossible to parallelize. Transformers were introduced to fix this issue with an encoder/decoder structure [53]. To understand the encoder/decoder intuitively, consider the problem of machine translation. If we have a sentence in written in Spanish, the encoder will attempt to construct a mathematical representation for the meaning of the sentence. The decoder will take this mathematical representation, as well as information about the English language (for example), and combine the two to create an English sentence. The meaning of the sentence and information about the English language are captured using a technique referred to as "attention" [53]. The following description of attention closely follows the source paper [53], and is provided for convenience.

In attention, an output is computed using a weighted sum of values, but with weights learned from a function that finds the compatibility between a query and the key corresponding to a value, where the query, the key-value pairs, and the output are all represented by vectors [53]. Attention is mathematically described as

$$\text{Attention}(K, V, Q) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V,$$

where K is the matrix containing the key vectors with d_k number of rows, V is the matrix containing the value vectors, and Q is the matrix containing the query vectors [53]. The scalar d_k is introduced as a normalization factor, lest dot-products become so large as to be unusable [53]. Different items are used as keys, values, and queries depending on the context. In the most basic case, the query is the word currently being examined, the key vector is all words being used as context for the query word, and the value vector is also all the words being used as the context for the query word. The output of the softmax function in the above equation is used as a weighting matrix for the value vectors comprising V.

It is often desirable for different weights to be learned based on some number, h, of different features of text, so the notion of "multi-head" attention is defined as

$$\text{MultiHead}(Q, K, V) = [\text{head}_1, \ldots, \text{head}_h] \cdot W^O,$$

where for each $j \in \{1, \ldots, h\}$,

$$\text{head}_j := \text{Attention}(Q \cdot W_j^Q, K \cdot W_j^K, V \cdot W_j^V),$$

with weight matrices W^O, W_j^Q, W_j^K, and W_j^V to be learned. Each head is empirically constructed to focus on different aspects of the training text [53]. These multi-head layers are stacked and then fed into a feed-forward neural network to form the encoder and decoder.

Transformers have been used in many state-of-the-art NLP models, such as GPT [46], BERT [7], and ERNIE [60]. In developing our conformal predictors, we choose to incorporate pre-trained word embeddings from BERT in particular. We focus on the use of "BERT-base" rather than "BERT-large" due to the high computation cost associated with the latter. Nonetheless, both deliver state-of-the-art results, so any minor trade-off in accuracy is justified. BERT-base has 12 layers, 768 hidden states, and 12 self-attention heads for a total parameter count of 110 million [7].

The main difference between BERT and the original transformer is its ability to examine context in both directions simultaneously, whereas the original transformer [53] and GPT [46] both gated the decoder layer, only allowing it to look in the direction from which it was supposed to be predicting. This proved effective, giving both versions of the original BERT state-of-the-art results across all generalized language understanding evaluation (GLUE) [56] tasks when the paper was published in 2019 [7]. BERT was pre-trained using two tasks, next sentence prediction (NSP) and MLM. In NSP, BERT is presented with two sentences and attempts to determine whether or not they are truly sequential. In MLM, BERT is presented with a masked word and asked to predict it given a context. During pre-training, 15% of words were masked so as to not let the model look at the correct answer while predicting. BERT was trained over the entirety of Wikipedia (approximately 2.5 billion words) and the BooksCorpus [62] in efforts to mimic language as closely as possible. A new sub-field, "BERTology", has surfaced in an attempt to explain why the embeddings are so efficient and generalizable [47]. We hope our application of conformal predictors to the BERT MLM task will contribute to this area of study.

2.3 Conformal Predictions

Conformal prediction uses knowledge gained from training a model to create confidence sets with guaranteed finite sample control over the probability of a type 1 error [48] and can be built on almost any machine learning tool, including neural networks [55]. Precisely, assuming exchangeable data examples, for any level of significance $1-\epsilon$ with $\epsilon \in (0,1)$, a conformal predictor yields a set-valued prediction with the property that it will fail to include the true label with probability at most ϵ [48]. This property, referred to as "validity," is mathematically guaranteed to hold for any finite sample size, but it is possible that the conformal prediction set is very large. The values included in the prediction sets are based on the "strangeness" of the test data when compared to training data, and the efficiency (i.e., size of the prediction sets) is dependent on how the strangeness measure – a so-called "nonconformity function" – is defined [55].

The only necessary assumption for the validity of conformal prediction sets is that the data must be exchangeable: a more relaxed assumption than the common assumption of independent and identically distributed, essentially meaning that for observed data examples $z_1, \ldots, z_n$, each of the $n!$ possible orderings of the examples were equally probable for being observed [48]. In that case, the collection of observed examples are best described by a "bag"

$$B := \wr z_1, \ldots, z_n \int,$$

denoting a collection of values such that the order of the elements is irrelevant [55]. For example, $\wr 1, 2, 2 \int = \wr 2, 1, 2 \int$.

A nonconformity measure A is a real-valued function that measures how strange or different a value z is from the other examples in the bag B. For the example values $z_i \in B$ for $i \in \{1, \ldots, n\}$, denote the nonconformity scores by

$$\alpha_i := A(B \backslash \{z_i\}, z_i). \tag{2.1}$$

The particular form of A is context/application-specific, but common choices include various norms, such as the ℓ_∞ norm in [32] or the ℓ_2 norm [48], of distances from a "center" of the set $B \backslash \{z_i\}$ to the point z_i.

Next, to decide whether to include a test value z in the conformal prediction set $\Gamma^\epsilon(z_1, \ldots, z_n)$ with level of significance $1-\epsilon$, first denote $z_{n+1} := z$ and update:

$$B := \wr z_1, \ldots, z_n, z_{n+1} \int .$$

Then, noting that α_{n+1} corresponds to the test value, include $z = z_{n+1} \in \Gamma^\epsilon(z_1, \ldots, z_n)$ if

$$p := \frac{|\{i = 1, \ldots, n+1 : \alpha_i \geq \alpha_{n+1}\}|}{n+1} > \epsilon.$$

This procedure is formally described in [55, 48] as a transductive conformal algorithm, and we summarize it here as Algorithm 1.

Throughout the remainder of the paper we will use the following notation. Let D denote a corpus of text, where the index $i \in \{1, \ldots, n\}$ denotes the position of the i-th word and n denotes the total number of words in D. For training, testing, and calibration, the entire corpus D is randomly split into three pieces D_{train}, D_{test}, and D_{cal}, respectively.

For many machine learning applications, however, transductive conformal prediction would be too computationally expensive since it requires recomputing all of the nonconformity scores for every new test observation/value. Motivated by this issue, ICP [39] is a modification of conformal prediction that greatly reduces computation costs. In ICP, the data is first split into proper training, calibration,

Algorithm 1: Transductive conformal algorithm.

Input: Nonconformity measure A, significance level ϵ, observations of feature-label pairs $z_1 = (x_1, y_1), \ldots, z_n = (x_n, y_n)$, and a new feature-label pair $z = (x, y)$.

Decide whether to include y in the set $\Gamma^\epsilon(z_1, \ldots, z_n, x)$.

$z_{n+1} := z$;

$B := \wr z_1, \ldots, z_n, z_{n+1} \int$;

for $i \in \{1, \ldots, n+1\}$ **do**

  $\alpha_i := A(B \backslash \{z_i\}, z_i)$;

end

$p := \frac{|\{i=1,\ldots,n+1 : \alpha_i \geq \alpha_{n+1}\}|}{n+1}$;

Include y in $\Gamma^\epsilon(z_1, \ldots, z_n, x)$ if $p > \epsilon$;

and testing sets D_{train}, D_{cal}, and D_{test}, as in our notation. Next, nonconformity scores are computed for the calibration set examples analogous to equation (2.1); letting $K := \{i \in \{1, \ldots, n\} : d_i \in D_{\text{cal}}\}$, the nonconformity score for every $j \in K$ is

$$\alpha_j := A(D_{\text{train}}, d_j).$$

Similarly, the nonconformity score for a test observation $d^* \in D_{\text{test}}$ is defined as $\alpha^* := A(D_{\text{train}}, d^*)$, and $d^* \in \Gamma^\epsilon$ if

$$p := \frac{|\{j \in K : \alpha_j \geq \alpha^*\}| + 1}{|K| + 1} > \epsilon.$$

Thus, the ICP algorithm must only be applied once to the calibration set, and each subsequent test value only requires calculating a single new nonconformity score to compare to the static collection of nonconformity scores in the calibration set. While ICP is slightly less reliable empirically than the transductive approach, the small sacrifice in empirical reliability does not outweigh the added benefit in computational efficiency [39]. From this point forward, any reference to conformal prediction should be interpreted as ICP unless otherwise stated.

Algorithm 2: ICP POS Prediction.

Result: Returns the conformal prediction set Γ^ϵ containing POS labels for a test word $d^* \in D_{\text{test}}$ and significance level ϵ.

$K := \{i \in \{1, \ldots, n\} : d_i \in D_{\text{cal}}\}$;

train the model using D_{train} to produce $\{\hat{y}_i : i \in K\}$;

for j *in* K **do**

  $s := y_j$; # Recall y_j is the true masked POS

  $\alpha_j := 1 - \hat{y}_{j,s}$;

end

for s *in* S **do**

  $\alpha_s^* := 1 - \hat{y}_{*,s}$;

  $p_s := \frac{|\{j \in K : \alpha_j \geq \alpha_s^*\}| + 1}{|K| + 1}$;

  if $p_s > \epsilon$ **then**

    $s \in \Gamma^\epsilon$;

  end

end

return Γ^ϵ;

Algorithm 3: ICP MLM.

Result: Returns the conformal prediction set Γ^ϵ containing candidate words for a masked token $d^* \in D_{\text{test}}$ and significance level ϵ.

$\widetilde{K} := \{i \in \{1, \ldots, n\} : d_i \in D_{\text{cal}}$ and d_i is masked$\}$;

train the model using D_{train} to produce $\{\hat{y}_i : i \in \widetilde{K}\}$;

for j *in* $\widetilde{K}$ **do**

  $u := y_j$; # Recall y_j is the true masked token

  $\alpha_j := 1 - \hat{y}_{j,u}$

end

;

for u *in* U **do**

  $\alpha_u^* := 1 - \hat{y}_{*,u}$;

  $p_u := \frac{|\{j \in \widetilde{K} : \alpha_j \geq \alpha_u^*\}| + 1}{|\widetilde{K}| + 1}$;

  if $p_u > \epsilon$ **then**

    $u \in \Gamma^\epsilon$;

  end

end

return Γ^ϵ;

3. METHODOLOGY

In this section we present our methodological contributions, namely ICP algorithms with nonconformity measures for POS tagging based on BERT and BiLSTM neural networks (both described by Algorithm 2), and for MLM based on a BERT neural network (described by Algorithm 3). Throughout this section, we overload the variable y: In POS tasks, y_i represents the true POS for the i-th word in D, whereas in the MLM tasks, y_j represents the true masked word for the j-th sentence in D, for $j \in \{1, \ldots, k\}$ where k is the total number of sentences in D.

3.1 POS Prediction

POS prediction involves finding the context of a word and then outputting the corresponding POS. Here we present our ICP Algorithm 2 for POS prediction. Let S represent the set of all unique POS in D, and for the i-th word in D, let $\hat{y}_i \in [0,1]^{|S|}$ represent the softmax vector produced by one of our two POS models, namely the subsequently described BERT POS (BPS) model or the BiLSTM model. In addition, let $\hat{y}_{i,s}$ denote the specific softmax value for any POS $s \in S$.

The nonconformity measure $\alpha_j = 1 - \hat{y}_{j,s}$ used in Algorithm 2 (and later, in Algorithm 3) represents the deviation from assigning softmax probability 1 (i.e., highest sensitivity) to the true label $s = y_j$ for the j-th word in the calibration set. Accordingly, the collection $\{\alpha_j\}_{j \in K}$ represents the distribution of the deviations in sensitivity that are consistent with the assigned softmax probabilities for the true

POS labels in the calibration set; if the noncomformity score associated with some POS $s \in S$ for a given word in the test sets falls in the tail of the distribution of $\{\alpha_j\}_{j \in K}$, then this is evidence at level ϵ suggesting that s is *not* a likely POS for the given test set word. Moreover, the nonconformity score $\alpha_j = 1 - \hat{y}_{j,s}$ is consistent with the general form of nonconformity scores commonly used for neural networks [see Chapter 4.2 of 55].

3.1.1 BERT POS Prediction

BERT creates custom embeddings for words based on the words themselves and the context around them. These embeddings can be fine-tuned to specific NLP tasks, such as POS prediction. We extend these predictions to form conformal prediction sets to quantify prediction uncertainty. The parameters of BERT that we implement for POS prediction have been pre-trained and are available from [7]. However, we must adjust the BERT parameters in addition to the parameters of a dense feed-forward network that we construct for mapping the BERT-base length 768 output embedding for a word to our $|S|$ component softmax vector [7].

There is some nuance to how we format the data to be usable with BERT. First, we address the BERT tokenizer. BERT separates a word root from its tense, but the practitioner must choose whether the root or the tense will be assigned a POS tag. We choose to assign the tag to the last token of a word (e.g., the word "wanted" is tokenized as "want" and "##ed", which are given the POS tags of [PAD] and [VBD] (verb past tense), respectively). This is so that BERT is able to identify the tense of the POS tag for prediction.

Second, it is necessary for a BERT input to have a fixed length of input tokens per batch. We chose to split sentences into token sequences of size 100 (i.e., each sentence was split at every 100th consecutive token). If the last sequence of the sentence was of length less than 100, we padded it with dummy [PAD] tokens to attain the desired length. The choice of a length of 100 tokens was a compromise due to the computational demands of fine-tuning BERT. Moreover, since most of the sentences in the Brown Corpus consist of less than 100 tokens, this truncation should have minimal effect on our results.

On top of BERT, we place a single softmax layer which reduces the 768 length vector into a $|S|$ length probability vector. Our model is trained by inputting a sentence and each word has its fine-tuned embedding vector run through the dense layer. We train the parameters for 3 epochs using the binary cross entropy loss with the RADAM optimizer [29]. A schematic illustration of our BERT architecture it is given in Figure 2. The softmax output vector from this neural network is then used in Algorithm 2 to yield the resulting conformal prediction sets. This combined BERT architecture with the conformal prediction algorithm for POS tagging is what we refer to as our BPS model.

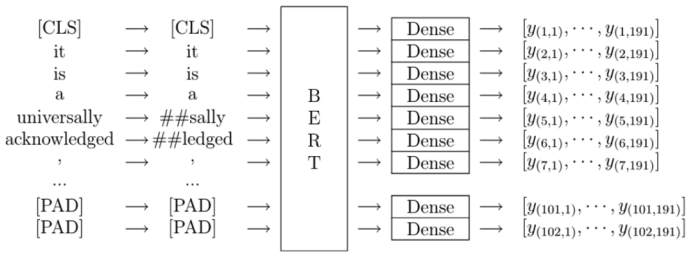

Figure 2: Illustration of the BERT POS model. The left most layer is the input sentence which is then transformed into the last token of each word. This 2nd layer is then input into BERT and an optimized embedding for the POS is made for each word. Each embedding is passed through a single layer dense neural net with sigmoid and softmax activation to produce the probability of each POS tag for each word in the sentence.

3.1.2 BiLSTM POS Prediction

In addition to our BPS model, we also construct a BiLSTM architecture for the task of POS tagging with conformal prediction sets, also using Algorithm 2. For word embeddings, we use Stanford's GloVe embeddings [42]. The GloVe embeddings are desirable because of their ability to balance local and global relationships between words. To make the model more generalizable, we chose to use pre-trained embeddings. Specifically, we use the GloVe embeddings which are of length 300 and trained on 6 billion tokens from Wikipedia and Gigaword [40]. Any word in our corpus that does not have a defined, pre-trained GloVe embedding is instead represented by a 300 length zero vector.

To train the BiLSTM model, we first create sentence embeddings to represent all of the sentences in our corpus. We create these sentence embeddings by concatenating the ordered, pre-trained GloVe word embeddings for the words in a given sentence. Accordingly, the sentence embedding for the j-th sentence is a matrix of dimension $300 \times n_j$, where n_j is the number of words in the j-th sentence. These sentence embedding matrices are then passed through a layer in the BiLSTM model. The BiLSTM layer consists of two sub-layers, a forward LSTM layer and a backward LSTM layer. For any individual sentence indexed by j, the forward LSTM layer takes in the matrix of embeddings and returns a matrix of dimension $150 \times n_j$. Similarly, the backward LSTM layer takes in the reversed matrix of embeddings and returns a matrix of dimensions $150 \times n_j$. Each column in these returned matrices contains a 150 length embedding suited for predicting the respective POS for each word. The idea is that the forward layer is capturing the context of a sentence that is processed from beginning to end, while the backward layer is capturing the context of a sentence that is processed from end to beginning. This extra context allows for the model to get a better understanding of the sequential patterns of POS in sentences. To combine the

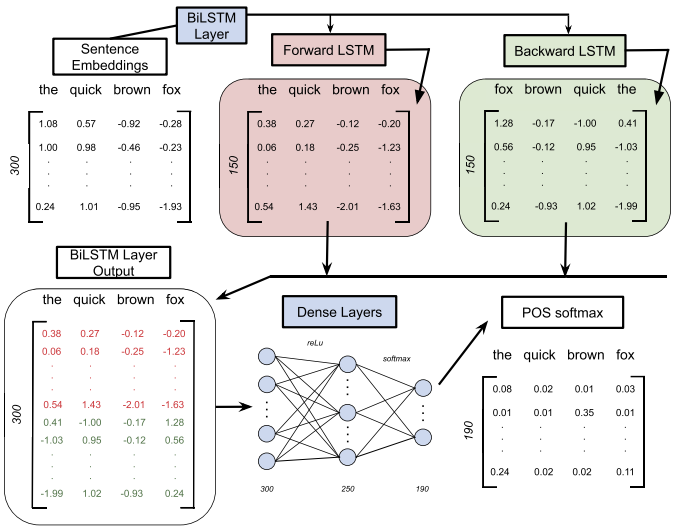

Figure 3: Illustration of the BiLSTM POS model processing a sample sentence embedding matrix. The top row illustrates the functionality of the BiLSTM layer within the model, with the leftmost matrix in the second row symbolizing the output of the BiLSTM layer. As seen, this output is simply the concatenation of the output matrix for forward LSTM layer and the reversed output matrix for backward LSTM layer. The rest of the second row provides a visualization of the dense layer processes which eventually result in the POS softmax matrix shown in the bottom right.

information gathered by the forward LSTM layer and the backward LSTM layer, we reverse the order of the columns of the matrix that were returned by the backward LSTM and concatenate it with the matrix that was output by the forward LSTM. This results in a $300 \times n_j$ matrix, with each column representing an optimal embedding for predicting POS.

After training the BiLSTM matrix of optimal embeddings, we pass the columns of this matrix through a feed-forward neural net. This net reduces the 300 length embedding to a 250 length vector with a ReLU activation, which is further reduced to a $|S|$ length softmax vector corresponding to the $|S|$ POS labels. Each softmax output vector represents an estimated probability distribution over the POS labels for a given word. This procedure is repeated for the n_j columns in the input matrix (each column corresponding to a word in the input sentence). The schematic for this BiLSTM architecture is displayed in Figure 3.

For training the parameters, we implement exponential decay in the popular RADAM optimizer [29]. We train for 700 epochs to avoid overfitting and we use cross entropy as our loss function. Finally, similarly to our BPS model, the softmax output vector from this neural network is then used in Algorithm 2 to yield the resulting conformal prediction sets.

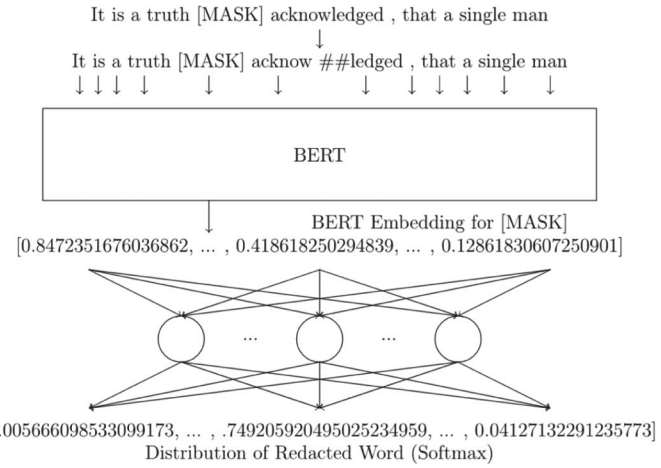

Figure 4: Illustration of the BERT MLM model. The top layer is the input sentence, which is then tokenized. A single token is then replaced with [MASK]. This tokenized sentence is then passed into BERT which outputs a softmax probability distribution corresponding to the masked token.

3.2 Masked Language Modeling

The MLM task is similar to POS tagging with two exceptions. First, the word to be predicted is masked or unknown (for training/testing, when a sentence is passed into the model, the target word is assigned the [MASK] token). Second, instead of classifying a word using $|S|$ POS labels, unknown words are inferred using a massive vocabulary of words. Though, these changes actually do not affect the basic conformal algorithm too much, as presented in Algorithm 3. From here on, "token" and "word" will be used interchangeably.

For MLM, we construct a BERT-based conformal prediction algorithm similar to the BPS model for POS tagging described in the previous section. BERT was designed for the task of predicting a masked word. Our BERT model takes the context and position of a [MASK] token and returns a softmax distribution over the 30,522 candidate tokens, and then Algorithm 3 is implemented to construct the conformal prediction set of candidate tokens for a given masked word of interest. Within Algorithm 3, U denotes the set of all 30,522 unique tokens comprising the set of pre-defined BERT tokens. For the j-th masked token in D, $\hat{y}_j \in [0,1]^{30,522}$ represents the softmax vector for the MLM model. In addition, $\hat{y}_{j,u}$ denotes the specific softmax value for any token $u \in U$. A schematic of our BERT MLM is given in Figure 4.

4. EMPIRICAL RESULTS

Using the Brown Corpus, we evaluate the conformal prediction sets produced by our three algorithms. The Brown Corpus contains 500 documents, with each word in these documents having a corresponding POS label. In total, there are just over 57,000 sentences and around 49,800 unique

words. We consider each sentence in the corpus as a data instance and randomly allocate 80% of these sentences for training, 10% for calibration, and 10% for testing. To account for sampling variability, the random allocation of the data into training, calibration, and testing sets is repeated 5 times, and all metrics are evaluated on and averaged over the 5 test sets.

For all POS tags (and combination of POS tags) we remove the hyphenated portion (if any). This includes headline (-HL), title (-TL), and emphasis (-NC) hyphenations, as well as foreign word prefix (FW-). If a word has a POS listed as a combination of multiple POS, the specific multiple POS combination is added as a new unique POS to our label set. After these preprocessing steps there remain $q = 190$ unique POS tags in the label set.

We consider the Brown Corpus because it has comprehensive, human-labeled POS tags. Further, the Brown Corpus has a significantly larger number of individual POS tags than most modern datasets, and is one of the only hand-tagged large corpora. These are valuable features for the development of our methods, and allow for better representations of how language is actually used. Moreover, as we illustrate in our real data example in Section 5, training on the Brown Corpus still yields reasonable performance for modern language applications. Finally, while the pre-trained BERT model is no longer new, it is still a very powerful algorithm even when compared to its more recent peers. Furthermore, there exists a large amount of reliable literature on the performance of BERT, which is important for evaluating the relative performance gains of our proposed ICP-enhanced BERT algorithms.

4.1 Performance Metrics

We consider a variety of metrics that evaluate both the "forced" point-predictions and the conformal prediction sets. The metrics we consider are adopted from the criteria considered in [32]. Let $n_{\text{test}} := |D_{\text{test}}|$, and assume a fixed $\epsilon \in (0, 1)$. For ease of notation, let $\hat{y}_i$ denote a prediction for some label y_i, for some example indexed by i. The metrics are defined as follows.

Classification accuracy (CA) is taken simply to be the proportion of correct predictions:

$$CA = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} I[\hat{y}_i = y_i].$$

Average credibility ($\overline{Cred}$) is the average maximum significance level ϵ required such that the prediction sets are nonempty:

$$\overline{Cred} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \sup\{\epsilon \ : \ |\Gamma_i^\epsilon| \geq 1\}.$$

The credibility of the i-th test point is the largest level ϵ (i.e., the largest type 1 error rate) such that the prediction set Γ_i^ϵ contains at least one label. Accordingly, low credibility is an indication of little confidence in any label. The OP criterion (for *observed perceptiveness*) is the average of all test p-values for correct classifications:

$$OP = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} p_{y_i}.$$

Conversely, the OF criterion (for *observed fuzziness*) is the average of all test p-values for incorrect classifications:

$$OF = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \sum_{y \neq y_i} p_y.$$

Average empirical coverage ($\overline{Coverage}$) is the proportion of prediction sets that contain the true value:

$$\overline{Coverage} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} I[y_i \in \Gamma_i^\epsilon].$$

Proportion of indecisive sets (PIS) is the proportion of sets (for a fixed ϵ) that contain more than one label:

$$PIS = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} I[|\Gamma_i^\epsilon| > 1].$$

The average confidence of decisive sets ($ACDS$) is the proportion of confidence sets of size 1 that contain the true label:

$$ACDS = \frac{\sum_{i=1}^{n_{\text{test}}} I[|\Gamma_i^\epsilon| = 1, y_i \in \Gamma_i^\epsilon]}{\sum_{i=1}^{n_{\text{test}}} I[|\Gamma_i^\epsilon| = 1]}.$$

Lastly, the N_ϵ criterion is the mean size of prediction sets at level of significance $1 - \epsilon$:

$$N_\epsilon = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} |\Gamma_i^\epsilon|.$$

4.2 POS prediction Results

Figure 5 and 6 present the results for both POS models. The metrics in Figure 6 require a forced point-prediction, which we take to be the label that maximizes the softmax vector that is returned by either the BPS model or the BiLSTM model.

It is observed in Figure 5 that for the 99% nominal confidence level, both models produce sets that average around 1–2 POS per set. This illustrates that the conformal prediction algorithm produces efficient sets at high confidence levels, and also suggests that the softmax probability vectors from the underlying neural nets are highly concentrated on 1–2 POS labels. Moreover, the $\overline{Coverage}$ and $ACDS$ values in Figure 5 demonstrate that these conformal prediction sets achieve their nominal coverage. Excessively small values for PIS with $N_\epsilon \approx 1$ at the 95% confidence level indicate a

	Proposed Conf.	99.9%	99%	95%
BiLSTM	$\overline{Coverage}$	0.9989	0.9903	0.9502
	$ACDS$	0.9996	0.9939	0.9631
	PIS	0.5424	0.1566	0.0024
	N_ϵ	3.4336	1.2732	0.9889
BPS	$\overline{Coverage}$	0.9990	0.9897	0.9499
	$ACDS$	0.9992	0.9909	NA
	PIS	0.3577	0.0334	0.0000
	N_ϵ	2.6260	1.0378	0.9570

Figure 5: Set-value prediction criterion results for POS prediction.

Model	CA	$\overline{Cred.}$	OP	OF
BiLSTM	0.9536	0.5055	0.5012	0.0493
BPS	0.9793	0.5020	0.5008	0.0126

Figure 6: Forced-value prediction criterion results for POS prediction.

high proportion of conformal prediction sets containing zero or one POS label.

To offer further insight, Figure 7 displays histograms of the set sizes for both models at the 99% confidence level. Many of the sets are of size one, which accounts for the height of the leftmost bins. However, the sizes of the sets vary greatly for different levels of nominal confidence, and so the uncertainty quantification afforded by the conformal prediction sets has utility. In particular, the models we constructed are able to provide 99.9% confidence for 3–4 POS labels, on average, for a given word. Such a quantified guarantee about the uncertainty in a prediction is not possible to provide from neural network architectures alone.

To demonstrate the validity for values of $\overline{Coverage}$ at more levels than the 99.9%, 99%, and 95% levels displayed in Figure 5, Figure 8 plots the average empirical coverage of the conformal prediction sets against their nominal levels for levels of significance ranging from 0 to 1. It is observed in Figure 8 that the solid and dotted lines are close together, as expected; this signifies that our prediction sets for both the BiLSTM and the BPS achieve approximately the desired amount of coverage.

Next, Figure 6 provides an assessment of the forced point-predictions of the underlying BPS and BiLSTM models. Being the state-of-the-art, it is found that the BPS model is marginally more accurate with respect to CA. However, both models perform relatively similar with regard to the other metrics in Figure 6. The difference in values between OP and OF indicate that the models are able to discriminate the correct POS label from the incorrect labels, on average.

Lastly, for further assessment of the conformal prediction algorithm, we present histograms of the nonconformity scores for the calibration sets in Figures 9 and 10. With

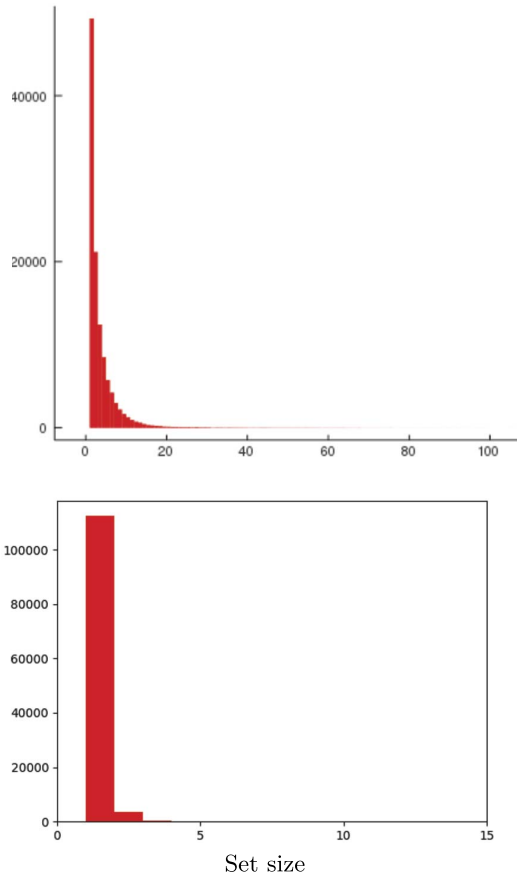

Figure 7: Histograms of conformal prediction set sizes for POS prediction at the 99% confidence level for BiLSTM (top) and BPS (bottom).

respect to the distributions of nonconformity scores, BPS produces smaller values than BiLSTM—as expected since transformers are a more complex model. When restricting the horizontal axes to $(0, 0.0002)$, the discrepancies in the distributions for predictions with low nonconformity scores become more evident. It is possible that the larger mass of the BiLSTM nonconformity scores near zero explains its higher ACDS.

4.3 MLM Results

For the MLM task, we mask a randomly chosen single word in each sentence in the Brown Corpus. Sentences are tokenized according to the "WordPiece" embeddings used by BERT, then truncated to a length of 128 to feed into the model. Further, we include fewer examples in the calibration set for the MLM task than in the previous section for the POS task due to the larger computational cost entailed by the much larger label set for MLM (i.e., all words in a vocabulary of around 30,000 words). Specifically, the calibration set contains around 1,300 sentences, and the testing set is also reduced to 1,000 sentences. To account for sampling

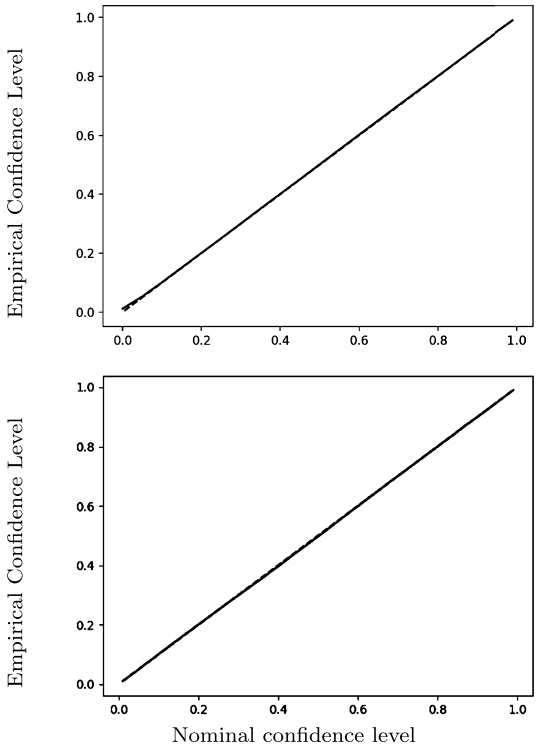

Figure 8: $\overline{Coverage}$ of conformal prediction sets for POS prediction for BiLSTM (top) and BPS (bottom). For reference, the dashed line is a 45 degree line.

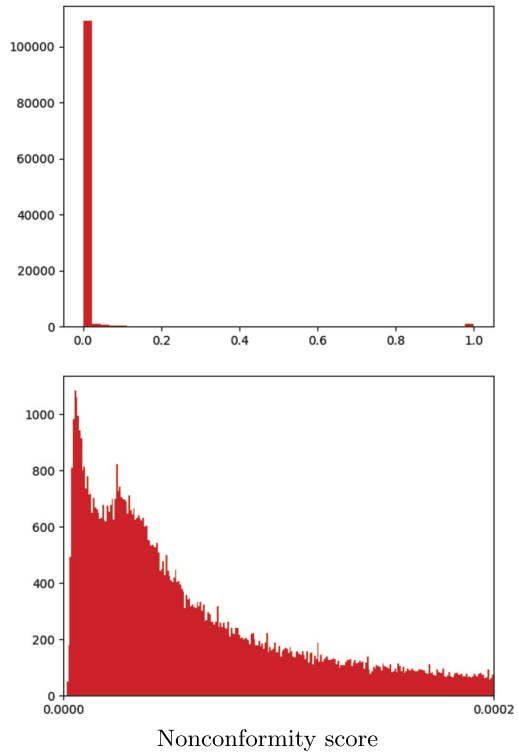

Figure 9: Histograms of nonconformity scores for the calibration sets for the BPS model. The histogram on the bottom plot only includes scores less than 0.0002 to better illustrate how these scores are distributed near zero.

variability in the random allocation of the data into training, calibration, and testing sets, we still repeat the process 5 times and report our results as averages of these 5 Monte Carlo iterations.

Unlike for POS prediction in the previous section, for MLM it is found that higher levels of confidence lead to prediction sets that are too large to be useful (see Figure 11). In particular, to guarantee that the true masked word is not omitted from the prediction set for more than 5% of test sentences (i.e., at the 95% level), the average conformal prediction set size is reported to be approximately 177 candidate tokens. Nonetheless, sacrificing some confidence quickly leads to smaller sets, down to 3–4 words on average at the 75% level. The histogram of the conformal prediction set sizes for all test examples is shown in Figure 13.

Additionally, the conformal prediction sets do achieve their nominal coverage at all levels displayed in Figure 11. To infer the validity for all values of $\overline{Coverage}$ from 75% to 95%, Figure 14 plots the average empirical coverage of the conformal prediction sets against their nominal levels of significance in this range.

Lastly, we provide the forced point-prediction metrics in Figure 12, and we present a histogram of the nonconformity scores for the calibration sets in Figure 15. The bimodal nature of the histogram is due to the underlying BERT model making overly discriminative predictions (i.e., the softmax vectors $\hat{y}_j$ being close to a one-hot vector), even when these predictions are sometimes very wrong, leading to either very high or very low nonconformity scores and not much in-between. Since the algorithm is picking a single word from a massive dictionary, we consider 54% CA to be reasonable, especially since we did not fine-tune BERT to our corpus. Moreover, many of these misclassifications are likely synonyms of the true word.

5. ILLUSTRATIVE REAL EXAMPLE

An application of our conformal prediction sets for MLM could come in the form of a post-hoc analysis tool for speech recognition software. The following example comes from a voice transcription of a 2009 TED Talk given by Michelle Obama, part of the greater TED-LIUM3 audio transcription corpus [19]. However, not all words were able to be detected by the automated speech recognition (ASR) system, and are instead labeled with the token <UNK> to take the place of the unknown word. Ideally, our model would be able to fill in these unknown words with set-valued predictions for any desired confidence level. To compare with other voice-to-text systems, we also analyzed the YouTube closed-captioning for this TED Talk video, which appeared to be more accurate than the ASR. Below are 3 example

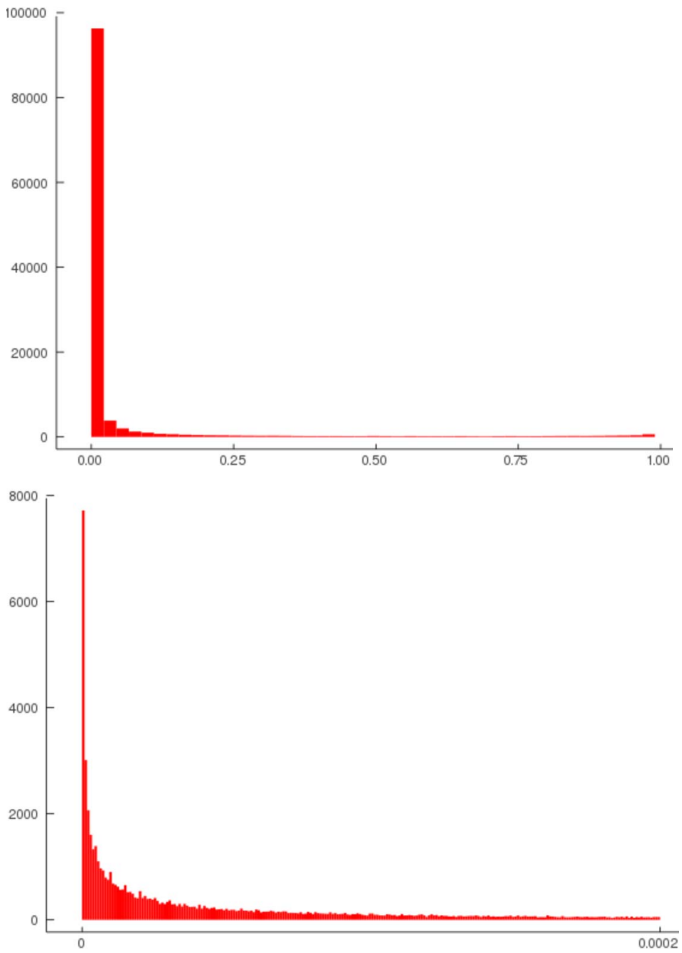

Figure 10: Histograms of nonconformity scores for the calibration sets for the BiLSTM POS model. The histogram on the bottom plot only includes scores less than 0.0002 to better illustrate how these scores are distributed near zero.

Confidence Level	$\overline{Coverage}$	N_ϵ
95%	.948	176.77
90%	.898	43.62
80%	.794	6.96
75%	.739	3.65

Figure 11: Set-value prediction criterion results for MLM.

Model	CA	$\overline{Cred}$	OP	OF
BERT MLM	0.542	0.609	.491	.122

Figure 12: Forced-value prediction criterion results for MLM.

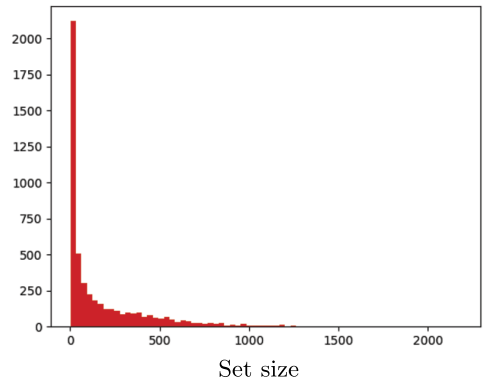

Figure 13: Histogram of conformal prediction set sizes for MLM at the 95% confidence level.

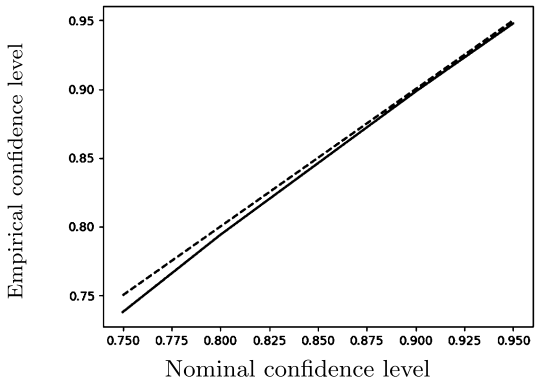

Figure 14: $\overline{Coverage}$ of conformal prediction sets for MLM. For reference, the dashed line is a 45 degree line.

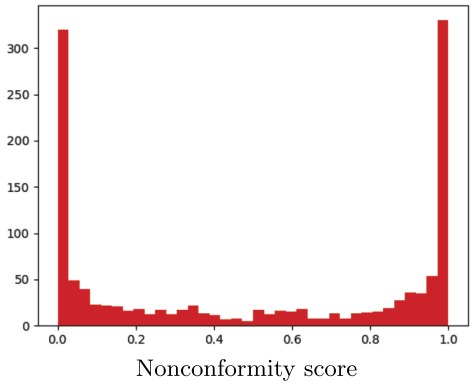

Figure 15: Histogram of nonconformity scores for the calibration sets for MLM.

sentences from the talk, with the italicized text representing the YouTube closed-captioning transcriptions, and the non-italicized text representing the ASR system transcriptions. The correct words, along with conformal prediction sets at the 75% confidence level (i.e., $\epsilon = 0.25$), are presented next.

Example 1.

> *. . . to go with him to a community meeting. But when we met, Barack was a community organizer.*
>
> . . . to go with him to a community <UNK>. But when we met, Barack was a community organizer.

$\Gamma^{0.25}$ = ['college', 'center', 'event', 'conference', 'meeting', 'dinner', 'gathering']
Correct word: 'meeting'

Example 2.

> *And he urged the people in that meeting, in that community, to devote themselves to closing the gap between those two ideas, to work together to try to make the world as it is and the world as it should be, one and the same.*
>
> And he urged the people in that meeting in that community to devote themselves to closing the gap between those two ideas, to work together to try to make the world as it is and the world as it should <UNK> one and the same.

$\Gamma^{0.25}$ = [',', 'be', 'seem']
Correct word: 'be'

Example 3.

> *And they opened many new doors for millions of female doctors and nurses and artists and authors, all of whom have followed them. And by getting a good education you too can control your own destiny.*
>
> And they opened many new doors for millions of female doctors and nurses and artists and authors all of whom have <UNK> <UNK>. And by getting a good education you too can control your own destiny.

$\Gamma_1^{0.25}$ = ['been', 'become', 'loved']
$\Gamma_2^{0.25}$ = ['children', 'died', 'success', 'experience', 'careers']
Correct words: 'followed', 'them'

At the 75% confidence level, the conformal prediction sets included the correct word in the first two examples. However, our MLM was not trained on any sentence with two consecutive masked words, thus it fails to include the correct words in the third example. That being so, if we pass this sentence through the model twice, each time with only one masked word, we see the more accurate results:

$\Gamma_1^{0.25}$ = ['joined', 'followed', 'loved', 'taught', 'inspired', 'influenced']
Correct word: 'followed'

$\Gamma_2^{0.25}$ = ['you', 'me', 'them', 'through', 'suit']
Correct word: 'them'

This suggests that the BERT model heavily depends on directly adjacent words to predict the token for a masked word in a sentence.

6. CONCLUDING REMARKS

We found that BERT-based conformal prediction sets were extremely effective in predicting both POS and masked words, which is unsurprising seeing as BERT is the dominant model for many NLP tasks at the moment. The complexity of models like BERT or BiLSTM was necessary, as our previous attempts using simpler nonconformity functions were not able to produce as efficient confidence sets. In the future, we may explore different nonconformity scores to get the BERT MLM prediction intervals even smaller. For example, scaling the noncomformity scores by a tuning factor or using a convex combination of the output for multiple models considered might lead to improved sensitivity and smaller prediction sets at a given level ϵ, where the tuning parameter or convex combination weights are trained via a loss function (such as PIS or average interval size) on a further validation set. Initial investigations show promising results, but these modifications are computationally more expensive than the methods described in our results section and remain a subject of future research.

ACKNOWLEDGEMENTS

The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Science Foundation nor the National Security Agency.

FUNDING

Research reported in this publication was supported by the National Science Foundationa and the National Security Agency under Award Numbers 2051010 and H98230-21-1-0014, respectively.

Accepted 18 August 2022

REFERENCES

[1] Bengio, Y., Simard, P. and Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* **5**(2) 157–166 (1994).

[2] Bohnet, B., McDonald, R., Simões, G., Andor, D., Pitler, E. and Maynez, J. Morphosyntactic tagging with a meta-BiLSTM model over context sensitive token encodings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* 2642–2652 (2018).

[3] Brants, T. TnT: a statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing* 224–231 (2000).

[4] Cauchois, M., Gupta, S. and Duchi, J. C. Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. *Journal of Machine Learning Research* **22**(81) 1–42 (2021). MR4253774.

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

[6] Devlin, J. and Chang, M.-W. Open sourcing BERT: state-of-the-art pre-training for natural language processing. *Google AI Blog* **2** (2018).

[7] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding (2019). arXiv preprint 1810.04805.

[8] Donahue, C., Lee, M. and Liang, P. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* 2492–2501 (2020).

[9] Elman, J. L. Finding structure in time. *Cognitive Science* **14**(2) 179–211 (1990).

[10] Fedus, W., Goodfellow, I. and Dai, A. M. Maskgan: Better text generation via filling in the _. In *International Conference on Learning Representations* (2018).

[11] Fisch, A., Schuster, T., Jaakkola, T. and Barzilay, R. Few-shot conformal prediction with auxiliary tasks (2021). arXiv preprint 2102.08898.

[12] Fisch, A., Schuster, T., Jaakkola, T. S. and Barzilay, R. Efficient conformal prediction via cascaded inference with expanded admission. In *International Conference on Learning Representations* (2020).

[13] Fisch, A., Schuster, T., Jaakkola, T. S. and Barzilay, R. Relaxed conformal prediction cascades for efficient inference over many labels. *International Conference on Learning Representations* (2021).

[14] Francis, W. N. and Kucera, H. Brown Corpus manual. *Letters to the Editor* **5**(2) 7 (1979).

[15] Gers, F. A., Schraudolph, N. N. and Schmidhuber, J. Learning precise timing with LSTM recurrent networks. *Journal of machine learning research* **3**. 115–143 (2002). https://doi.org/10.1162/153244303768966139. MR1966056.

[16] Giovannotti, P. and Gammerman, A. Transformer-based conformal predictors for paraphrase detection. In *Conformal and Probabilistic Prediction and Applications*, *PMLR* 243–265 (2021).

[17] Goldberg, Y. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research* **57**. 345–420 (2016). https://doi.org/10.1613/jair.4992. MR3584073.

[18] Heid, S. H., Wever, M. D. and Hüllermeier, E. Reliable part-of-speech tagging of historical corpora through set-valued prediction. *Journal of Data Mining and Digital Humanities* (2020).

[19] Hernandez, F., Nguyen, V., Ghannay, S., Tomashenko, N. and Esteve, Y. TED-LIUM 3: twice as much data and corpus repartition for experiments on speaker adaptation. In *International conference on speech and computer* 198–208. Springer (2018).

[20] Hochreiter, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **6**(02) 107–116 (1998).

[21] Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation* **9**(8) 1735–1780 (1997).

[22] Hu, Y., Huber, A., Anumula, J. and Liu, S.-C. Overcoming the vanishing gradient problem in plain recurrent networks (2018). arXiv preprint 1801.06105.

[23] Jurafsky, D. and Martin, J. H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* 3rd ed. (2021). https://web.stanford.edu/~jurafsky/slp3/.

[24] Koutník, J., Greff, K., Gomez, F. and Schmidhuber, J. A clockwork RNN. In *International Conference on Machine Learning*, *PMLR* 1863–1871 (2014).

[25] Kupiec, J. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech & Language* **6**(3) 225–242 (1992).

[26] Lafferty, J., McCallum, A. and Pereira, F. C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning 2001* (2001).

[27] Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L. and Luís, T. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* 1520–1530 (2015).

[28] Liu, D., Fu, J., Liu, P. and Lv, J. Tigs: An inference algorithm for text infilling with gradient search. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019).

[29] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J. and Han, J. On the variance of the adaptive learning rate and beyond. *International Conference on Learning Representations* (2020).

[30] Liu, L., Shang, J., Ren, X., Xu, F., Gui, H., Peng, J. and Han, J. Empower sequence labeling with task-aware neural language model. In *Proceedings of the AAAI Conference on Artificial Intelligence* **32** (2018).

[31] Maltoudoglou, L., Paisios, A., Lenc, L., Martínek, J., Král, P. and Papadopoulos, H. Well-calibrated confidence measures for multi-label text classification with a large number of labels. *Pattern Recognition* **122**, 108271 (2022).

[32] Maltoudoglou, L., Paisios, A. and Papadopoulos, H. BERT-based conformal predictor for sentiment analysis. In *Conformal and Probabilistic Prediction and Applications*, *PMLR* 269–284 (2020).

[33] Manning, C. D. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *International conference on intelligent text processing and computational linguistics* 171–189. Springer (2011).

[34] Messoudi, S., Rousseau, S. and Destercke, S. Deep conformal prediction for robust models. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* 528–540. Springer (2020).

[35] Mortier, T., Wydmuch, M., Hüllermeier, E., Dembczynski, K. and Waegeman, W. Efficient algorithms for set-valued prediction in multi-class classification (2019). arXiv preprint 1906.08129. https://doi.org/10.1007/s10618-021-00751-x. MR4277133.

[36] Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P. and Allen, J. A corpus and evaluation framework for deeper understanding of commonsense stories (2016). arXiv preprint 1604.01696.

[37] Olah, C. Understanding LSTM networks (2015). http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[38] Paisios, A., Lenc, L., Martínek, J., Král, P. and Papadopoulos, H. A deep neural network conformal predictor for multi-label text classification. In *Conformal and Probabilistic Prediction and Applications*, *PMLR* 228–245 (2019).

[39] Papadopoulos, H. Inductive conformal prediction: Theory and application to neural networks. In *Tools in Artificial Intelligence*, *IntechOpen* (2008).

[40] Parker, R., Graff, D., Kong, J., Chen, K. and Maeda, K. *English Gigaword fifth edition*. Linguistic Data Consortium, Philadelphia (2011). Technical Report.

[41] Pascanu, R., Mikolov, T. and Bengio, Y. Understanding the exploding gradient problem (2012). arXiv preprint 1211.5063.

[42] Pennington, J., Socher, R. and Manning, C. D. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* 1532–1543 (2014). https://doi.org/10.1126/science.aaa8685. MR3382218.

[43] Peters, M., Ammar, W., Bhagavatula, C. and Power, R. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* 1756–1765 (2017).

[44] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of NAACL-HLT* 2227–2237 (2018).

[45] Plank, B., Søgaard, A. and Goldberg, Y. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of ACL 2016*, *Association for Computational Linguistics (ACL)* (2016). https://doi.org/10.1162/COLI_a_00253. MR3553982.

[46] Radford, A., Narasimhan, K., Salimans, T. and Sutskever,

I. Improving language understanding by generative pre-training (2018).
[47] Rogers, A., Kovaleva, O. and Rumshisky, A. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics* **8**. 842–866 (2020).
[48] Shafer, G. and Vovk, V. A tutorial on conformal prediction. *Journal of Machine Learning Research* **9**(3) (2008). MR2417240.
[49] Sharfuddin, A. A., Tihami, M. N. and Islam, M. S. A deep recurrent neural network with BiLSMT model for sentiment classification. In *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)* 1–4. IEEE (2018).
[50] Shen, T., Quach, V., Barzilay, R. and Jaakkola, T. Blank language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* 5186–5198 (2020).
[51] Srinivasan, S., Arora, R. and Riedl, M. A simple and effective approach to the story cloze test. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)* 92–96 (2018).
[52] Sun, X. Structure regularization for structured prediction. *Advances in Neural Information Processing Systems* **27**. 2402–2410 (2014).
[53] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems* 5998–6008 (2017).
[54] Vovk, V., Gammerman, A. and Saunders, C. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning, ICML '99* 444–453. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1999).
[55] Vovk, V., Gammerman, A. and Shafer, G. *Algorithmic learning in a random world*. Springer (2005). MR2161220.
[56] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. and Bowman, S. R. GLUE: A multi-task benchmark and analysis platform for natural language understanding (2019).
[57] Wang, P., Qian, Y., Soong, F. K., He, L. and Zhao, H. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network (2015). arXiv preprint 1510.06168.
[58] Xin, Y., Hart, E., Mahajan, V. and Ruvini, J. D. Learning better internal structure of words for sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* 2584–2593 (2018).
[59] Yasunaga, M., Kasai, J. and Radev, D. Robust multilingual part-of-speech tagging via adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* 976–986 (2018).
[60] Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M. and Liu, Q. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* 1441–1451 (2019).
[61] Zhu, W., Hu, Z. and Xing, E. Text infilling (2019). arXiv preprint 1901.00158.
[62] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A. and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision* 19–27 (2015).

Neil Dey. North Carolina State University, United States. E-mail address: ndey3@ncsu.edu

Jing Ding. North Carolina State University, United States. E-mail address: jding7@ncsu.edu

Jack Ferrell. University of Florida, United States. E-mail address: jacksax290@gmail.com

Carolina Kapper. High Point University, United States. E-mail address: ckapper@highpoint.edu

Maxwell Lovig. Yale University, United States. E-mail address: max.lovig@yale.edu

Emiliano Planchon. North Carolina State University, United States. E-mail address: eplanch@ncsu.edu

Jonathan P. Williams. North Carolina State University, United States. E-mail address: jwilli27@ncsu.edu