Mapping Local News Coverage: Precise location extraction in textual news content using fine-tuned BERT based language model

Sarang Gupta *
Data Science Institute
Columbia University, NY
sg3637@columbia.edu

Kumari Nishu *
Data Science Institute
Columbia University, NY
kn2492@columbia.edu

Abstract

Mapping local news coverage from textual content is a challenging problem that requires extracting precise location mentions from news articles. While traditional named entity taggers are able to extract geo-political entities and certain non geo-political entities, they cannot recognize precise location mentions such as addresses, streets and intersections that are required to accurately map the news article. We fine-tune a BERT-based language model for achieving high level of granularity in location extraction. We incorporate the model into an end-to-end tool that further geocodes the extracted locations for the broader objective of mapping news coverage.

1 Introduction

A media or news desert is an uncovered geographical area that has few or no news outlets and receives little coverage. Mapping locations mentioned in news articles is the primary step in identifying news deserts. A key challenge in the process is to manually peruse the corpus of news articles, identify the location mentions and assign spatial coordinates which can then be placed on a map to identify a newsroom's coverage.

While conventional Named Entity Recognition (NER) taggers such as those offered by spaCy (Honnibal and Montani, 2017), Natural Language Toolkit (NTLK) (Bird et al., 2009) and Stanford NLP (Finkel et al., 2005) group contain tags to identify organizations, geo-political entities (GPE) and certain non-GPE locations such as mountain ranges and bodies of water from text, they are not able extract precise location mentions such as addresses, streets or intersections in their entirety. For example, the sentence - *"The family will hold shivah from 7 to 9 p.m. Thursday, Oct. 13, and again Saturday, Oct. 15, at Temple Sinai, 5505 Forbes Ave., Pittsburgh."* passed through the Stanford Named Entity Tagger, returns "Temple Sinai", 'Forbes Ave." and 'Pittsburgh" as separate location entities. However, to accurately map the location of interest, one requires the whole address - 'Temple Sinai, 5505 Forbes Ave., Pittsburgh.' to be returned as a single location.

In recent years, there has been an advent of powerful pre-trained deep learning based language models such as Google's Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018), XLNet (Yang et al., 2019) and OpenAI's GPT-2 (Radford et al., 2018). These models can be fine-tuned for specific classification tasks in the absence of abundant training data and thus are often helpful with weak supervision (Ratner et al., 2017). We fine-tune the BERT model to extract precise location mentions in their entirety as iterated in the previous paragraph. We first build a dataset of about 10,000 sentences extracted from a corpus of 80,000 news articles spanning Jan 2018 to June 2019 published in Philadelphia Inquirer newspaper. We use Amazon Mechanical Turk (MTurk) to label the locations of interest in the sentences and fine-tune the BERT based NER tagger to classify the words in a text. Finally, we incorporate geocoding of the extracted locations into the pipeline.

In this work, we present an end-to-end system to extract geographic data from text. This tool could be particularly useful for newsrooms to map the coverage of their printed content helpful in identifying news deserts or by researchers and other organizations to extract precise location mention in text. Our main contributions are as follows: (1) Preparing a dataset containing sentences with words tagged as geopolitical entities, organizations, streets and addresses. (2) Fine-tuning an existing BERT based NER tagger to identify the aforemen-

* Equal contribution

Proceedings of the Fourth Workshop on Natural Language Processing and Computational Social Science, pages 155–162

https://doi.org/10.18653/v1/P17

tioned location entities. (3) Building an end-to-end system that consumes news content, transforms into BERT readable format and returns a list of the geocoded locations mentioned in the content.

The overall process is illustrated in Figure 1. We will first discuss our modelling approach followed by the process that we followed for geocoding the extracted locations.

2 Related Work

Named Entity Recognition (NER) is a well studied area in the field of Natural Language Processing (NLP). It aims to identify different types of entities such as people, organizations, nationalities and locations in text. Tools trained on conventional NER models such as Conditional Random Fields (Lafferty et al., 2001), Maximum Extropy (Ratnaparkhi, 2016) and LabeledLDA (Ramage et al., 2009) have been successful in identifying common named entities. However, challenge comes when high level of granularity is of interest in extracting location entities such as specific addresses, streets or intersections.

Lingad et al. (2013) evaluated the effectiveness of existing NER tools such as Stanford NER, OpenNLP and Yahoo! PlaceMaker on extracting locations from disaster-related tweets. Brunsting et al. (2016) presented an approach combining NER and Parts-of-Speech (POS) tagging to develop a set of heuristics to achieve higher granularity for location extraction in text.

There has been some work around the use of end-to-end neural architecture on several sequence labelling tasks including NER (Chiu and Nichols, 2015) and POS (Meftah and Semmar, 2018) tagging. Magnolini et al. (2019) explored the use of external gazetteers for entity recognition with neural models showing that extracting features from a rich model of the gazetteer and then concatenating such features with the input embeddings of a neural model outperforms conventional approaches.

Fine-tuning pre-trained language model for domain-specific machine learning tasks has become increasingly convenient and effective. Lee et al. (2019) introduced BioBERT, a BERT based biomedical language representation model for biomedical text mining and Xue et al. (2019) presented a fine-tuned BERT model for entity and relation extraction in Chinese medical text. Liu (2019) presented advances in extractive summarization using a fine-tuned BERT model.

To our knowledge, no previous work has been done to fine-tune a pre-trained language-based deep learning model to achieve the level of precision and granularity in location extraction that is required for the purpose of mapping news coverage. Furthermore, there does not exist an open source end-to-end tool to extract and geocode precise locations from a piece of text.

3 Model

Our approach to developing a named-entity tagger for the task of precise location extraction involves fine-tuning an existing neural network on a target dataset. Fine-tuning a model updates its pre-trained parameters, improving its performance on the downstream NLP task.

We treat the task of named-entity tagging in a sentence as that of token classification within a sequence, assigning each word (token) in the sentence (sequence) a label. The fine-tuning process for token classification involves: (1) Preparing the training dataset with expected labels for tokens within each sequence (2) Loading an existing model with pre-trained weights (3) Extending the model with a classification layer at the end with number of nodes equal to the number of classes in the task at hand (4) Training the model on the target dataset.

We will use Bidirectional Encoder Representations from Transformers (BERT), developed by Google AI in 2018 as the pre-trained model. BERT makes use of multiple multi-head attention layers to learn bidirectional embeddings for input tokens. It is trained for masked language modeling, where a fraction of the input tokens in a given sequence are masked and the task is to predict the masked word given its context (Devlin et al., 2018). Our decision to choose BERT is motivated by the fact that it is a general purpose language representation model pre-trained on millions of articles on English Wikipedia and BookCorpus. Given the diversity of topics present on these two training sets, we believe BERT would be able to generalize well to our dataset containing news articles. BERT's use of WordPiece tokenizer mitigates the out-of-vocabulary issue while tokenizing location names which are often proper nouns. With minimal architecture modification, BERT can be applied to our NER task.

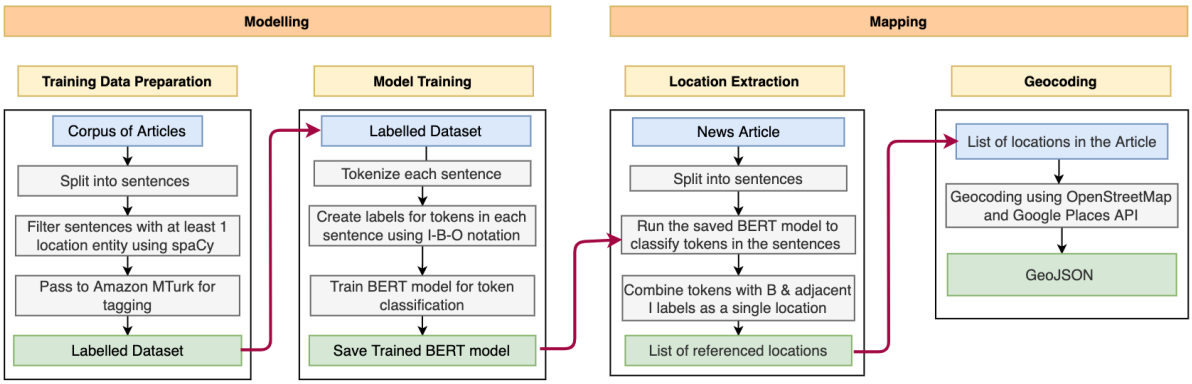

Figure 1: Overview of our methodology

3.1 Dataset Preparation

Our goal is to create a labelled dataset for token classification with each token labelled as being part of a location entity or not. For this purpose, we assembled a dataset of 10,000 sentences drawn from a corpus of 80,000 news articles ranging from January 2018 to June 2019 published in the Philadelphia Inquirer newspaper. The articles represented 'beats' including politics, opinions, sports, food and travel among others, published by a number of different authors. The articles originated from 10 different news sources which had their articles published on Philadelphia Inquirer website.

Since, majority of the sentences in the articles did not contain a location mention, sending the whole article for tagging on Amazon MTurk was not cost-efficient. To overcome this, we broke down the articles into individual sentences using spaCy's Sentencizer and devised a set of heuristics and custom rules on top of spaCy's NER system to capture sentences with mentions of addresses, streets and intersections. Through exploratory analysis and manually perusing the articles, we identified three patterns based on the syntactic relation between POS and NER tags present in the sentence:

Heuristic 1.

$$NER_{LOC} + POS_{PREP} + NER_{LOC}$$

Two location entities separated by a prepositional tag often highlighted a hierarchical location between two location entities. For example: In the phrase *'Arbor Street in Kensington.'* (Figure 2 - Top), refers to the street - Arbor Street, which is part of the neighborhood - Kensington.

Heuristic 2.

$$NER_{NUM} + POS_{NOUN} + POS_{PREP} + NER_{LOC}$$

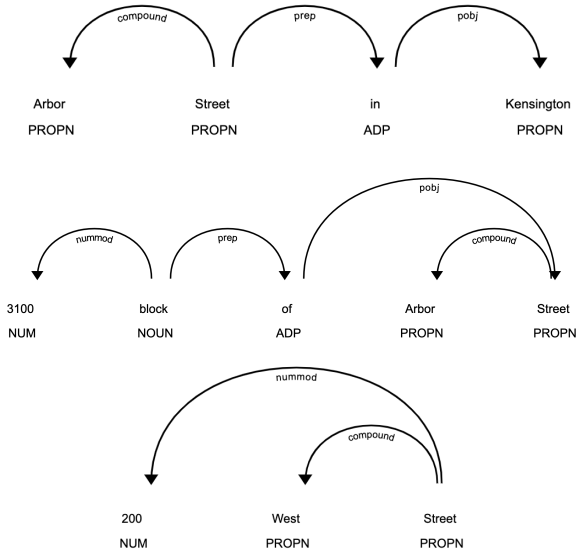

Figure 2: Heuristics to identify locations of interest

A number, noun and a prepositional tag preceding a place were collectively used to identify a precise location in the referenced place. For example, *'3100 block of Arbor Street.'* (Figure 2 - Middle)

Heuristic 3.

$$NER_{NUM} + POS_{LOC}$$

A number preceding a location entity, often a street, collectively referred to a building in a street or area. For example, *'200 West Street'* (Figure 2 - Bottom) refers to a building on the West Street.

Using the aforementioned, we filtered in sentences that contained at least one of the above patterns. Note that even though the defined heuristics are not exhaustive and can not define all possible syntactical patterns in which our locations of interest could exist in text, they offer an effective

strategy to create labelled data for higher-level, less precise supervision required in transfer learning (Ratner et al., 2019). We augmented the dataset to include 20% of sentences that contained simple geo-polical entities (such as United States, Pennsylvania, Philadelphia etc.) as it is easier for a pre-trained language model to identify such mentions. A total of 10,000 sentences were randomly selected from this set.

As a second pass, the crowd workers on Amazon Mturk were instructed to identify locations of interest in the 10,000 sentences and mark their starting and ending character numbers in the sequence. The sentences were then tokenized using WordPiece tokenization (Wu et al., 2016) and the tokens were assigned labels using Inside-outside-beginning (I-O-B) notation (Ramshaw and Marcus, 1995). For instance, LOC_B represents the starting of a location entity and LOC_I represents subsequent tokens that are part of that location entity. An example is depicted in Figure 3 which shows tokens in a sentence tagged using the I-O-B notation.

3.2 Model Training

We used BERT's implementation provided by Hugging Face (Wolf et al., 2019) in PyTorch (Paszke et al., 2019). As the task was framed as a multi-label classification problem, a softmax layer comprising of 6 nodes (LOC_B, LOC_I, X, O, SEP, CLS) was added for token-level classification. The weights were initialized using BERT pre-trained for general purpose entity recognition (Link to GitHub repository of the pre-trained model). We fine-tuned both BERT_{LARGE} and BERT_{BASE} and used the original cased vocabulary of the respective models.

The models were trained using the BERTAdam (Adam (Kingma and Ba, 2014) optimizer with weight decay regularization for BERT) optimizer, with learning rate set to 3×10^{-5}. A weight decay rate of 0.01 to the main weight matrices alongside early stopping was used to add regularization. The average length of WordPiece tokenized sequence in our dataset was 40 (max: 241, min: 12, std: 16); we used a maximum sequence length of 128 for BERT_{Base} and 64 for BERT_{Large}. A batch size of 32 was used for BERT_{Base} and 16 for BERT_{Large} and the models were trained for a total of 20 epochs. NVIDIA Tesla K80 and NVIDIA Tesla P100 GPUs on the Google Cloud Platform were used to fine-tune BERT_{Base} and BERT_{Large} respectively. Due to computational limitations, we could not train BERT_{Large} with larger sequence lengths and batch sizes.

3.3 Experimental Results

In order to ensure the effectiveness of our experiment, we divided the dataset into training, development and test sets to maintain a ratio of 8:1:1. As the task was framed as a multi-label classification problem, we calculate common performance measures - Precision, Recall and F_1 scores (Liu et al., 2014) for each of our labels (LOC_B, LOC_I and O). The results are presented in *Table 1*.

Table 1: Results on the Test Set for BERT_{Base} and BERT_{Large}

		Metric		
Model	Tag	Precision	Recall	F_1
BERT_{Base}	LOC_B	76.97	87.43	81.85
	LOC_I	74.83	71.97	73.38
	O	99.25	99.10	99.18
BERT_{Large}	LOC_B	78.27	81.01	79.62
	LOC_I	73.01	70.16	71.56
	O	98.14	98.22	98.18

As shown in *Table 1*, BERT_{Base} performs better than BERT_{Large} for all tags on all three evaluation metrics. We believe this is due to the fact that we used a shorter sequence length while training the BERT_{Large} model, which ignores any location mentions after token 64.

To assess the overall performance of our model, we calculated the average precision, recall and F_1 scores weighted by the number of LOC_B and LOC_I labels in the test set. As there were a total of 2334 tokens tagged with the label LOC_B and 3355 tokens tagged with LOC_I in the test set, using weighted average scores helps in accounting for this imbalance. We obtain a weighted average F_1 of 76.83 on BERT_{Base} and 74.87 on BERT_{Large} (Table 2). Note that we have used a conservative approach in measuring the performance of the models. While all tokens might not be essential in identifying a location, our evaluation metrics requires the beginning (LOC_B) and all subsequent tokens (LOC_I) to be identified for the location to be marked as correct.

Figure 4 illustrates the comparison between our fine-tuned BERT_{Base} model and standard NER tools such as those offered by spaCy and Stanford

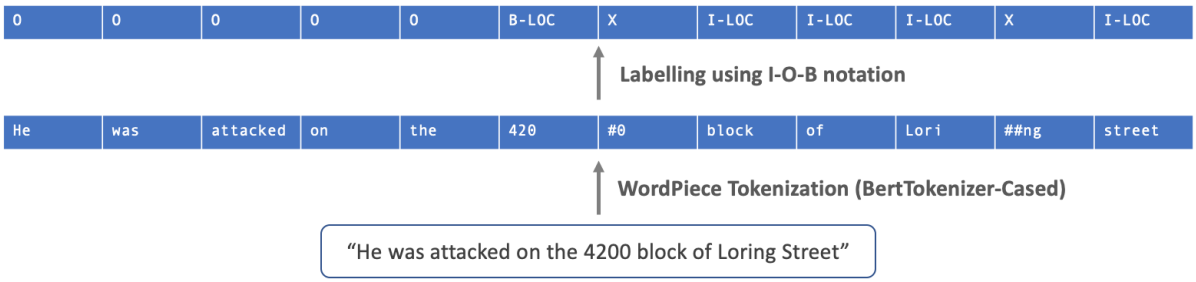

Figure 3: WordPiece Tokenization and Tagging

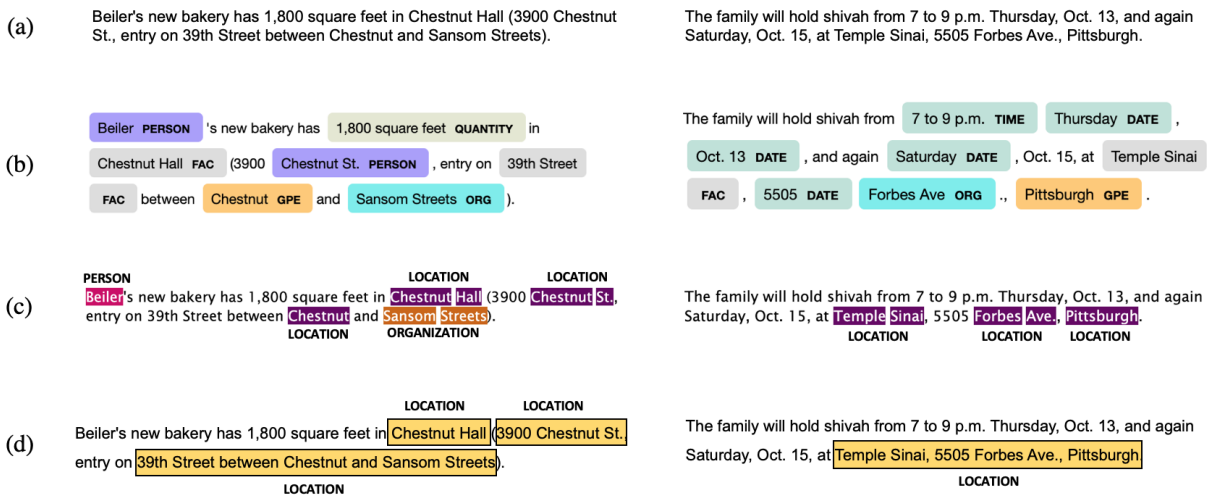

Figure 4: Illustration of different NER taggers (a) Original Sentence (b) SpaCy NER (c) Stanford NLP Group (d) Fine-tuned BERT$_{Base}$ Note: Desciption of SpaCy NER tags can be found at: https://spacy.io/api/annotation

Table 2: Weighted Average Scores on the Test Set for BERT$_{Base}$ and BERT$_{Large}$

	Metric (Weighted Average)		
Model	Precision	Recall	F_1
BERT$_{Base}$	75.70	78.28	76.83
BERT$_{Large}$	75.17	74.62	74.87

NLP group. Our model is able to extract precise location entities whereas the other tools split the entities into sub-entities.

4 Mapping

By deploying the fine-tuned BERT model, we present a system that (1) consumes textual news content (or any other text for that matter), (2) extracts the raw text of the locations referenced and (3) returns the corresponding geocodes. The geocodes can be plotted on a map using any standard mapping tool.

4.1 Location Extraction

The BERT models in (3) have been fine-tuned to perform entity recognition at a sentence level. Hence, the first step in location extraction is to split the content into individual sentences. Similar to the approach taken in the data preparation step, we use spaCy's Sentencizer to break the content into individual sentences. The sentences are then passed through the fine-tuned BERT model which return a list of precise location mentions in the sentence. The locations are regrouped at the content level.

4.2 Geocoding

Geocoding is the process of taking input text, such as an address or the name of a place, and returning a latitude/longitude location on the Earth's surface for that place. For instance, geocoding "Atlantic City" will yield 39.3643, 74.4229.

The locations are gecoded using geocoding APIs offered by Google Places and OpenStreetMap. The

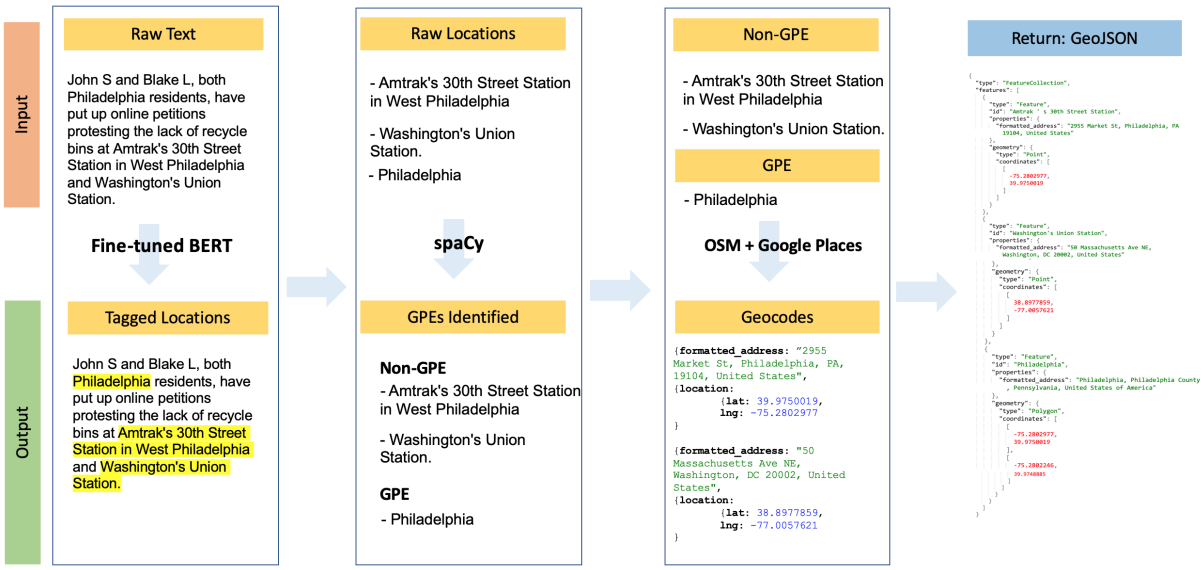

Figure 5: An example output of the mapping system

Text Search Service within the Google Places API returns information about a set of places based on a string, which makes it particularly useful for ambiguous address queries. However, due to limited number of free queries available every month, we utilize the open source OpenStreetMap API in conjunction with Google Places API to minimize the cost associated with geocoding. The OpenStreetMap API does not have an in-built text disambiguation service like Google Places' TextSearch but it is able to return geocodes for geo-political entities with high accuracy. To this extent, we use OpenSteetMap for geocoding locations which are geo-political entities and Google Places for other locations. Locations are identified as geo-political by passing the entity list through spaCy's NER tagger.

To facilitate disambiguation, we utilize the *location* and *radius* parameters in Google Places API and the *viewbox* parameter in the OpenStreetMap API which allows users to specify the preferred region of search. This is particularly useful for mapping local news coverage which is generally confined to a single region. Geo-political entities are geocoded as polygons with series of coordinates defining the enclosed area. For other locations a single point coordinate representing the centroid is returned. The final output is a GeoJSON containing a list of all extracted locations with their geocodes which can readily be consumed by a third-party mapping software.

In Figure 5, we can see an example of a input text and the corresponding GeoJSON output of the locations referenced in the text.

5 Conclusion and Future Work

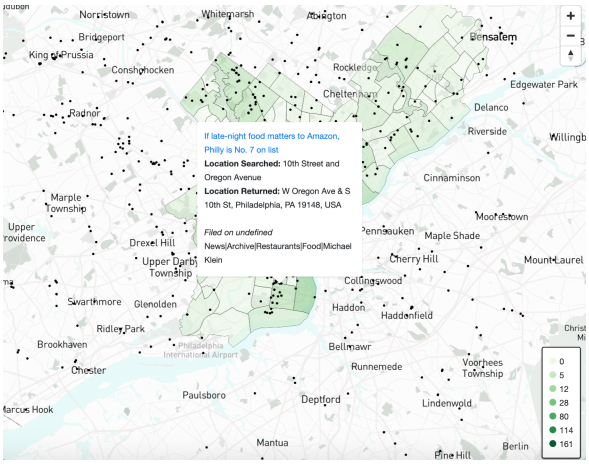

Figure 6: 1000 articles from Philadelphia Inquirer plotted on the map. The green polygons represent Philadelphia neighborhoods with color indicating number of times they have been referenced. Dots represents the locations referenced. The tooltip shows the metadata for the article that references one of the locations.

To map local news coverage, it is important to extract precise location mentions from textual news content. In this paper, we presented a fine-tuned BERT based language model to achieve high level of granularity that is required for this task. Compared to traditional NER taggers, our model is able to extract locations such as addresses, streets and intersections in their entirety, making it possible to

accurately place them on a map. We also present an end-to-end system by deploying the model to extract locations from textual content and geocode them in a format that can be directly consumed by a mapping software for plotting.The system can be integrated with an interactive user interface to visualize location-related features of news content. An example is illustrated in Figure 6 of 1000 random news articles from Philadelphia Inquirer plotted using Mapbox API. The map shows the locations referenced in these news articles (denoted by the points) and the coverage across different Philadelphia neighborhoods (green polygons with color indicating the frequency).

Our work can be advanced and extended from many different perspectives. First, a more comprehensive dataset could be developed with specific tags for different location types. The BERT model can be further fine-tuned using this dataset to extract different location types from the text. Second, the disambigation of the extracted location can be further strengthened using contextual clues to enhance the accuracy of the geocoding process. Lastly, many other state-of-art pre-pretrained language models can be fine-tuned using our dataset and a comparison can be established to select the best performing model to be used for tagging.

Acknowledgments

We would like to thank the Brown Institute for Media Innovation at Columbia University and the Lenfest Institute for Journalism for providing us the resources to carry out this study and for their guidance throughout.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Shawn Brunsting, Hans De Sterck, Remco Dolman, and Teun van Sprundel. 2016. Geotexttagger: High-precision location tagging of textual documents using a natural language processing approach.

Jason Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-snns. *Trans. Assoc. Comput. Linguist.*, 6.

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

Jenny Finkel, Trond Grenager, and Christoper Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

John Lingad, Sarvnaz Karimi, and Jie Yin. 2013. Location extraction from disaster-related microblogs.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *CoRR*, abs/1903.10318.

Yangguang Liu, Yangming Zhou, Shiting Wen, and Chaogang Tang. 2014. A strategy on selecting performance metrics for classifier evaluation. *International Journal of Mobile Computing and Multimedia Communications*, 6:20–35.

Simone Magnolini, Valerio Piccioni, Vevake Balaraman, Marco Guerini, and Bernardo Magnini. 2019. How to use gazetteers for entity recognition with neural models. In *Proceedings of the 5th Workshop on Semantic Deep Learning (SemDeep-5)*, Macau, China. Association for Computational Linguistics.

Sara Meftah and Nasredine Semmar. 2018. A neural network model for part-of-speech tagging of social media texts. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256, Singapore. Association for Computational Linguistics.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning.

Adwait Ratnaparkhi. 2016. *Maximum Entropy Models for Natural Language Processing*, pages 1–6. Springer US, Boston, MA.

Alex Ratner, Paroma Varma, Braden Hancock, and Chris Ré. 2019. Weak supervision: A new programming paradigm for machine learning.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel. *Proceedings of the VLDB Endowment*, 11(3):269–282.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Kui Xue, Yangming Zhou, Zhiyuan Ma, Tong Ruan, Huanhuan Zhang, and Ping He. 2019. Fine-tuning bert for joint entity and relation extraction in chinese medical text.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.