A REPRESENTATION BASED MULTIPLE INSTANCE LEARNING FRAMEWORK FOR GEOLOCATION OF TEXT

by

Sneha Nagpaul
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
(Computer Science)

Committee:

____________________ Dr. Huzefa Rangwala, Thesis Director

____________________ Dr. Jessica Lin, Committee Member

____________________ Dr. Aditya Johri, Committee Member

____________________ Dr. Sanjeev Setia, Chairman, Department of (Computer Science)

____________________ Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: ____________________ Spring 2018
George Mason University
Fairfax, VA

A Representation Based Multiple Instance Learning Framework for Geolocation of Text

A thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science at George Mason University

By

Sneha Nagpaul
Bachelor of Science
University of Delhi, 2010

Director: Dr. Huzefa Rangwala, Professor
Department of (Computer Science)

Spring 2018
George Mason University
Fairfax, VA

Acknowledgments

Foremost, I would like to extend my gratitude to my advisor and thesis director, Dr. Huzefa Rangwala for his motivation, enthusiasm, and immense knowledge. This project could not have been completed without his patience and guidance during the process. I am particularly appreciative of the focus he provided to this work in a rapidly evolving research environment.

In addition to my advisor, I would like to thank the rest of my thesis committee: Dr. Jessica Lin and Dr. Aditya Johri for their encouragement, time and insightful comments.

I would also like to acknowledge Mohammad A. Rahman, who helped me navigate the murky waters of research and helped point out further differences that enhanced the novelty of this work. A special thanks to Kyle Jackson for reading the draft and providing notes on language.

Finally, I would like to thank my family and friends.

Table of Contents

List of Tables

List of Figures

Abstract

A REPRESENTATION BASED MULTIPLE INSTANCE LEARNING FRAMEWORK FOR GEOLOCATION OF TEXT

Sneha Nagpaul

George Mason University, 2018

Thesis Director: Dr. Huzefa Rangwala

Given the deluge of data caused by crowd generated content from social media websites, the complexity of extracting information from has increased manifold. An important characteristic of such text is its original location which can in turn be used to respond to emergencies such as floods and crimes[1]. The patterns discovered by such geolocation of social media related unstructured text can also be used commercially for targeted advertising and recommender systems[2][3][4]. This work deals with geolocating text from Twitter data that are labeled with a user's information. However, instead of locating the user who can be viewed as a collection of tweets, it focuses on locating individual tweets. For this task, the problem is described within the multiple instance learning framework and a novel approach using neural networks is designed which trains a tweet level classifier using only user location labels. The model outperforms the state of the art in multiple instance learning and provides significant scalability and speedup compared to existing method. Superseding the Bag of Words models from prior geolocation research, the intuitive instance level neural network classifier discovers high level language features such as grammar and identifies name places without feature engineering[5][6].

Chapter 1: Introduction

As the content generated by social media platforms grows, it is becoming increasingly complex to distill information from it. A simple yet useful characteristic that can be extracted from the text based data is the location of the content generator at a given time. This information can be leveraged to encode events that are actionable for authorities such as natural calamities and crimes[7][1]. Additionally, learning the location specific patterns in this language can be used for commercial purposes such as targeted advertising, resource allocation and recommender systems[2][3].

It is often the case in real world scenarios that labeled information is available at an aggregated level for a set of instances; however, knowledge distillation is required at a more granular level. An example of a nefarious use of these methods could be if the outcome of an election in an area is known, the probability of each of the voters' choices could be determined; thus, threatening their privacy[8]. However, a use case that furthers social good can be found in medicine where given the genetic makeup and the occurrence of a disease(label), individual genes can be held responsible and targeted for treatment and diagnosis. In the language processing community, researchers have worked on the problem that if the sentiment of a product or movie review is known, how this sentiment will be propagated to the sentence or the word level[9]. The method that is proposed here can be viewed as an application of the multiple instance framework. In this work, given discourse from a certain region (higher level), the method will endeavour to transfer this regional label to a language substructure like a phrase.

Prior work on predicting location of conversation tries to classify substantially lengthy texts using indicative words into regional groups. In particular, research has focused on location labels available for a user active on Twitter[5][10]. This body of work distorts phrasal information in language to identify words pertaining to a certain region by using a Bag of words model to process language.

Viewing this problem within the multiple instance learning framework can help to solve the geolocation problem at a granular level while preserving the structure of the text. In this formulation each tweet can be regarded as an instance and each user can be regarded as a location group, also called labeled bag containing all tweets by this user. The multiple instance learning model then propagates the label at the user level to each tweet which is most indicative of the users location.

Prior work in multiple instance learning makes strong assumptions about the membership of instances inside a bag in order to train the classifier based on the bag label[11]. More recently, researcher have tried to relax this assumption by assuming that bag level labels are simply an aggregation of instance level predictions[9]. However, even these relaxed methods rely on a similarity measure that requires training of deep neural networks in order to learn meaningful embeddings for classification which can be viewed as de facto feature engineering. A common issue with prior approaches is the use of a similarity kernel which makes the methods hard to scale as the size of the dataset grows.

In this work, the problem is set up as finding patterns of language that represent a certain geographic location when aggregate level information is provided and needs to be transferred to each instance to locate an individual tweet.

1.0.1 Contributions

In this work, a novel framework for Multiple Instance Learning is proposed using neural networks which addresses the issues of prior research in the respective areas and makes the following contributions:

1. Provides a framework for transferring bag labels to instance level predictions using a representation learning framework which can accommodate any input directly without the need to encode with embeddings or kernel. This is a major contribution of the work since it eliminates the need for subjective feature engineering and learns the

latent space directly from the labeled data.

2. Model is trained using gradient descent through an end-to-end trainable neural network. Since the model training can work with batches, the proposed method scales up easily in terms of memory requirements and provides a significant computational speedup.

3. Relaxes assumptions about instance level memberships in bags, while providing a flexible setting to train an instance level classifier. This increases applicability by providing an intuitive architecture for finding instance level classifier which can be theoretically replaced with any trainable neural network architecture of required complexity. This flexibility is also extends to how the instance level labels are aggregated to form bag level labels.

4. The proposed model tests itself on a novel application for a geolocation task that classifies tweets to a particular location and thus discovers language indicative of a region.

1.0.2 Thesis Outline

The rest of this work is organized into 5 sections. Section 2 provides background and related research in fields of multiple instance learning and dialectology while pointing out to their contributions and gaps. Section 3 describes the method used to address the shortcomings of existing research and goes into detail on model architecture and training methods. Section 4 details the dataset, experimental setting and results and provides illustrative examples of the model. Section 5 concludes with a discussion and suggests future directions.

Chapter 2: Background and Related Work

Given this setup of the geolocation problem in this work, we discuss prior literature in Multiple Instance learning which deals with transferring of labels from a broader to a finer level and Geographic Information Retrieval which is used to map language to location in general. Finally, a brief review of neural networks is provided that introduces the capability of representation learning for discovering non-linear relationships in data without feature engineering.

2.1 Multiple Instance Learning

Multiple Instance Learning generally refers to a class of learning methods that work with labeled information at an aggregate level while trying to distill information at a disaggregate level. This natural framework is prevalant in a variety of problems and has been employed in solving content based image retrieval and classification [12], privacy related applications[13], categorising texts[11] and recognising objects[14].

This framework was traditionally employed to explore the feature space for a supervised learning problem, where many contenders of feature sets constituted the individual instances for the original instance (the bag) but only one of those feature vectors was responsible for the bag level labels[15]. Thus, the aggregation function was an OR relationship, wherein negative bags had only negative instances and positive bags had at least one positive instance[15]. This stringent assumption fails to generalize as soon as the use case moves away from exploring potential features for a certain supervised learning problem. Additionally, the main goal of this formulation was to infer bag level labels as opposed to exploring granularity[15].

In the context of locating a person from a tweet, someone belonging to a certain location might travel and tweet about the destination. It is also possible that they might relocate and absorb the linguistic style of the new region. Hence, the OR formulation is inappropriate for the location use case as all tweets may not be indicative of region. There might be a general language pattern to account for or travel which may result a negative bag to have positive instances.

In order to relax this stringent assumption, researchers have proposed other ways to link instance level labels to higher level labels. A work by Xu and Frank posits that there is equal contribution from all the constituents of a bag which are considered independent[16]. Zhou et al.[17] make the assumption that within a negative bag all instances are negative and then use a semi supervised approach to classify further. In another generalisation of the original aggregation, Weidmann et al. [18]classify bags based on the presence of a collection of types of instances where a type is a concept which can be viewed as an at-least-one-of-each formulation instead of an at-least-one-formulation. This would be akin to engineering features and looking for their presence in the user. As a hypothetical example, if they are health conscious and eat orgranic food and reference technology in their tweets then we might want to classify them as a belonging to San Francisco. This formulation, though intuitive, would require separate treatment of any datasets and would stand on strong assumptions made by the modeler as opposed to discovering patterns within the data.

Another body of work tries to shift focus from bag level to instance level predictions. In particular, Liu et al. [19], utilize this framework to identify key instances within a bag i.e. those that are most relevant to the positively labelled bags using nearest neighbour heuristics subject to various aggregation schemes. While this formulation is particularly fitting to the geolocation use case, the combinatorial complexity for the nearest neighbour based metrics and checking for various ways instances in the positive bags might have voted towards the bag label make this approach unscalable.

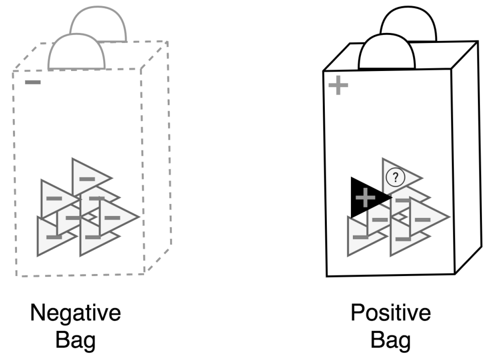

Figure 2.1: Multiple Instance Learning Framework - In the traditional setting, the negative bags(left) would only contain negative instances and the positive bags(right) would contain at least one positive instance. This is the implication of the OR function constraint on the instance level to bag level label transfer[15].

2.1.1 Single Instance Learning (SIL)

A naive way to tackle the problem is Single Instance Learning (SIL) which assumes that every instance of a bag has the bags label[20]. This assumption is also equivalent to an AND transfer relationship. This approach seems to match the framework of the problem under consideration as it is not an outlandish assumption that if one tweet is from New York, the rest are too. However, it doesnt take into account general phrases across various places and thus could confuse the classifier due to additional noise in the dataset[20].

2.1.2 Support Vector Machines for MIL

A more recent body of work harnesses the Support Vector Machine (SVM) framework while maintaining the instance to bag label transfer relationship as the traditional OR [11] [15]. In their work Adrews et al. [11] propose MISVM models which train the parameters using all possible combinations of instance and bag level labels to find the model that maximizes the soft margin criterion. A kernel based formulation can be seen in Gartner et al. [21], where traditional support vector machines are deployed to work on the bag level directly. In computer vision related applications, researchers have attempted to find regions of interest within an image by connection through instance level labels [22][23]. Sparse multi- instance learning approaches are mentioned in [24] and transductive SVM's [25] have been explored which cluster the region to find low density areas and then place the decision boundaries in those regions.

Thus, due to the presence of a kernel and testing for many possibilities, the computational complexity of support vector based methods combinatorially increases rendering them unscalable in terms of memory and computational resource needs. These drawbacks limit the use of MISVM to relatively smaller datasets.

2.1.3 Group Instance Cost Function (GICF)

More recently, neural networks are being leveraged in order to address the rigidity of aggregation functions[9]. The Group Instance Cost Function (GICF) method uses a cost function(Equation (2.1)) that tries to find the parameters optimized for instance and bag level costs using a general differentiator and a specific differentiator that uses a similarity measure to determine these differences.

$$J(\Theta) = \frac{1}{N^2}(\sum_{i=1}^{N}\sum_{j=1}^{N}\exp(-\|x_i - x_j\|_2^2)(\sigma(\Theta^\top x_i) - \sigma(\Theta^\top x_j))^2 + \frac{\lambda}{K}\sum_{k=1}^{K}((\frac{1}{|\mathcal{G}_k|})(\sum_{i\in\mathcal{G}_k}\sigma(\Theta^\top x_i) - l_k)^2 \tag{2.1}$$

GICF's use of a similarity measure makes it a kernel based method like SVMs which causes similar scalability issues. Another perspective on the similarity kernel is that there are certain requirements for feature engineering before the model can process the bags and instances. The generated feature vectors are then passed on to a training method with the two-tier cost function (2.1) to find a classifier at the bag level. Following this, the instances can be treated by the classifier as a bag of one to be classified. While this method takes into account modern deep learning architectures, it still requires designing of the similarity measure which can be subjective.

While GICF is the closest in terms of architecture to the work described in this paper, there are a few more things to consider while comparing the proposed method to it, particularly the problems these methods have originally tackled. GICF has been tested on sentiment data wherein the instances are sentences in a review and share heavy context. In case of a bag of tweets, even though the tweets come from the same person, they have no requirement to have been originated at the same time and share any context as is the case with a paragraph of a review.

GICF has been tested on sentiment analysis datasets consisting of considerable amount of information at the bag level where instances are generally more connected to each other contextually. However, for Twitter based datasets, these assumptions fall through and it is somewhat harder to transfer information to instance level as instances are not connected in the same temporal or contextual space.

MISVM and related works use strong assumptions for instance membership and aggregation functions. The work that relaxed these assumptions is GICF but requires hard to train embedding vectors and similarity measures. Old works have focused on context heavy instance bag combinations such as reviews. The proposed method will try to complement the prior research by applying the framework within the constraints of geolocation of twitter based dataset which consists of tweet instances that share the source but not the context necessarily. Thus relaxing assumptions of membership and aggregation enough for the data to be able to provide those patterns to the model.

2.2 Geographic Information Retrieval

Geographic Information Retrieval refers to a class of methods that deal with mapping language to location[26]. General tasks include geolocating text, essentially predicting the location for a given text and dialectology which deals with studying language patterns of a certain geographical area. This work extends GIR literature as it endeavours to predict the location of tweets.

2.2.1 Traditional Methods

The common theme in traditional GIR is an auxiliary dataset, called gazetteer, that maps words to locations statically[26][27]. In addition to this gazetteer lookup, heuristic based feature engineering methods such as term frequency analysis and pinpointing the positional attributes of words are leveraged to accomplish this task. These methods were traditionally used on regional applications due to the non-scalable nature of the gazetteer[28]. For application at a larger scale to online datasets, a naive implementation of place name detection followed by disambiguation was adopted to keep the size of the auxiliary dataset tractable[29].

2.2.2 Language Modeling

A scalable class of methods that modelled languages emerged as datasets became bigger and more intractable towards the early 2010s[30]. Unsupervised learning methods such as topic modelling were initially explored which are harder to scale[27]. To overcome the gazetteer disadvantage and achieve larger scalability, supervised learning methods were used on the datasets that are labelled with user level information[30]. In the context of this paper, they deal with user level labels and require a string of tweets to geolocate on the bag level.

2.2.3 Neural Networks for GIR

With the emergence of deep learning and neural networks, recent work has focussed on modelling language features with neural networks. Liu et al. [31] work at locating users instead of language using feedforward neural networks. Another recent work, uses bag of words representations which distorts structural language features of user documents for a neural network to classify and find indicative words[5].

A shortcoming of these methods is that they require a considerable amount of information even to geolocate at a bag level - they do not locate a tweet, they locate a user. Another issue is the extensive use of the bag of words model which distorts structural information in language that could be indicative of region. This work addresses both of these problems by leveraging representation learning at a fine grained level with minimal preprocessing. By generalising the design of the neural network, milNN also provides a framework to vary architecture choices based on expected complexity without feature engineering.

2.3 Neural Networks and Deep Learning Primer

Machine learning has conventionally been a process where researchers try to transform raw input data into an internal representation that is appropriate for a learning subsystem to consume in order to identify or classify patterns of interest[32]. Representation learning is a class of methods that deal with automatically finding these internal representations in order to perform machine learning tasks. Deep Learning is one kind of representation learning method that deal with finding these features through a series of nonlinear transformations on minimally processed input to discover representation that make the data suitable for tasks like classification and detection[32].

A key feature of these methods is that there is minimal intervention with the data before being fed to the model and there is no requirement for explicit feature engineering from experts in a particular field. For example, a linguist need not be involved to work on language related tasks. Hence, if the method was initially developed for english, it would just as easily generalize to other formal languages such as chinese or french and informal languages, such as those used on Twitter and Facebook and vice versa.

Another major advantage of deep learning over any methods that use feature engineering is that if the usage changes in the future, the model will simply update itself based on the newer pattern. This idea is known as concept drift, where the data inherently changes and older patterns become obsolete[32]. In the case where a conventional machine learning approach is being used, a new feature vector would need to be designed using considerable subjectivity and time of a domain expert in order to accommodate such changes manually.

2.3.1 Multilayer Neural Networks and Back-propagation

A simple yet powerful application can be found in the popular feed forward neural networks or multi layer perceptron (MLP) which map a fixed size input and to a fixed size output. The intermediate layers are a series of nonlinear activations on affine transformations of the inputs which are all trainable[32]. These intermediate weights are learned using backpropagation which is an application of the chain rule for composite functions wherein the error is propagated backwards from the point where a label is available for a particular training instance and a loss can be calculated and adjustments can be made at initial levels all the way to the first layer where the input was fed into the network[33]. The direction of change of the intermediate weights is determined by calculating the gradient which is derivative of the nonlinear transformation and the adjustment is made in the opposite direction of the gradient.

While deep learning methods have been widely accepted by the computer vision community, this review will focus on how language related applications have benefited from the neural network paradigm with exception of a few illustrative examples.

Due to the implications of the universal approximation theorem[34], a neural network with a single hidden layer is capable of generalising any function given enough data and time. While this theoretical results makes basic neural networks attractive, it is often faster to explore exotic architectures to accomplish these tasks in reasonable time with data restrictions. Hence, the rest of this section will discuss convolutional neural networks and recurrent neural networks.

2.3.2 Convolutional Neural Networks

Convolutional Neural Networks are a class of deep learning models that deal with spatial correlations in the input[35][36]. The models involve sliding a filter on some structured data that may have compositional structure and subsequently discovering motifs that may be present throughout the raw structure. An example of such a structure is an image that has two balls at opposite corners and a filter would detect the ball having strong local similarity yet being able to occur in another part of the picture. This filter is a mathematical convolution and was traditionally hand designed as a part of a feature engineering step (eg. Edge detection). However, with neural networks, these filters are learned as trainable weights which are shared within each feature map which is essentially the output of applying the filter to the previous layer. The non linearity can be introduced as an activation on the tensor generated as the feature map. While this formulation is intuitive and popular in vision related applications, more recently they have shown to be useful with language and sequence related tasks[37]. In these language related applications, the convolution can be seen as a one dimensional window being slid over language substructures looking for patterns.

2.3.3 Distributed Representations of Words

When shifting focus from the vision application to lanaugage the input space changes considerably. Now, as opposed to an image which is a tensor of numbers, symbols like words and letters require a distinct point of view for being tackled by a neural network. It is helpful to understand this handling with the example of an experiment where given a sequence of words the next word is the target for prediction as described in [38]. The input to this model is a one hot encoded series of words that belong to the sequence. At the first layer, every word in the series generates a pattern of activations or a word vector. This vector is then mapped by subsequent layers to the output which is a large vector that includes the probability of each of the words being next. While training, this model's output is the one hot encoded vector for the predicted word. While backpropagating from those errors, a distributed representation is learned by the activations in the input layer. LeCun et al.[32] talk of them as multiple micro-rules for symbols that the network automatically learns. This can also be viewed as 'intuitive' inference as opposed to logical inference that would stem from rigid grammatical rules.

One such example of these distributed representation is the Word2Vec model[39]. In the vector space, the word for king and queen appear closer whereas they would be far apart in a lexicon. Thus moving away from rigid representations to those that share meaning beyond fixes rules that need to manufactured by domain experts. Vector representations appear widely in natural language processing literature to accomplish a multitude of tasks [40][41][42][43][44][45][46]. Representations learned during training of a certain network can be indicative of how the language/symbols relate to the task. For example, the earlier experiment of predicting the next word provides spatial set up of language in the word vectors. Another perspective for looking at these vectors (or embeddings) is as a language model where the vector constitutes a particular word's role and usage instructions in that model's space. Older work goes as far as N-gram type modeling which is a generalisation of a bag of words model where N = 1. Those models do not have the capabilities of modeling relationships between words like the distributed representations do.

2.3.4 Recurrent Neural Networks

While MLPs and CNNs are successful neural network models, they exhibit rigidity in the input they can take in that the input to these feed forward neural networks must be of a constant length. Recurrent Neural Networks provide more flexibility to this input structure as they process the input sequence by the element and maintain an internal state with the information they have encountered[32]. A modification of RNNs build a longer context into the state vectors called Long Short Term Memory Models[47]. LSTMs consist of gated units that chose to remember or forget information at each time step. Thus, they are able to maintain states with older information than vanilla RNNs. These methods have been extensively employed in machine translation [45] along with processing other sequential inputs.

In contrast to RNN's which can be viewed as a deep feed forward neural networks with all layers sharing the weights[32], the method proposed in this paper also has sharing of weights, but in the width of the network rather than the depth.

Chapter 3: Proposed Method

3.1 Problem Statement

Given a user U_i with a binary location label $y_i \in \{0,1\}$ where 1 denotes that the user is from a particular city and 0 denotes otherwise. Each U_i is a collection of tweets t_{ij}, j= 1,2,...n and the task is then to devise a function $f(t) \rightarrow y$ which essentially labels individual tweets as belonging to the city under consideration.

3.2 Method Description

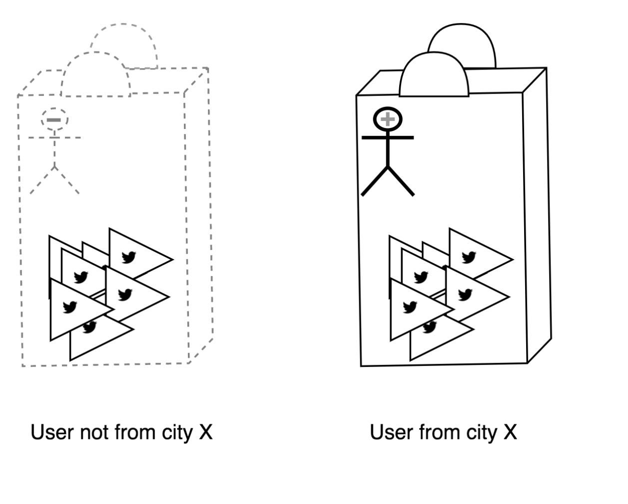

Figure 3.1: Multiple Instance Learning for Twitter- This figure shows how the Muliple Instance Learning framework is mapped to the current problem. Each user is a bag and is labeled negative or positive depending on whether they belong to a particular city. There instances are not assumed to belonging to certain classes given user/bag level labels but the aggregation function assumed here is and average.

Hence, as Figure 3.1 describes, a user is a bag labelled with location as being positive to belonging to a city or negative which the tweets form the instances in the multiple instance learning setting which are devoid of labels at the training stage. Thus, an instance level model is designed with trainable neural network components as though labels were available. This model is them simply plugged into the architecture described in Figure 3.2 along with an association function that can aggregate these labels to the bag level for loss discovery and traning through gradient descent.

For a treatment of the problem as formulated here, a fully trainable neural network architecture is proposed in this work and is called milNN. The model's architecture is described in Figures 3.2, 3.3 and 3.4.

Instance Level Classifier: (Figure 3.3) The tweet classifier is an embedding layer followed by a hidden layer that maps to a classifier. The model is regularised using dropout after the hidden layer.

Bag Level classifier: (Figure 3.4)This instance level classifier is then applied to N tweets and the results are averaged to get the bag level labels. This 'average' can be substituted by any assumption given the complexity of the relationship.

Loss function and Training Method: (Figure 3.1) The bag level loss is binary cross entropy loss (Equation (3.1) and it is back-propagated using Adam Optimizer [48] to learn the weights of the network. Thus, the instance level classifier gets trained as a result of the bag level losses being back-propagated.

$$\mathcal{L}(\hat{y}, y) = -\frac{1}{N}\sum_{i}^{N}[y_i log\hat{y_i} + (1 - y_i)log(1 - \hat{y_i})] \tag{3.1}$$

3.3 Method Characteristics

Due to being fully embedded in the neural network and representation learning paradigm, milNN doesnt have any explicit feature engineering requirements. Hence, it negates the immeasurably time consuming and subjective involvement of a domain expert. Additionally, it is equipped to handle any changes that might occur organically in the data as it veers

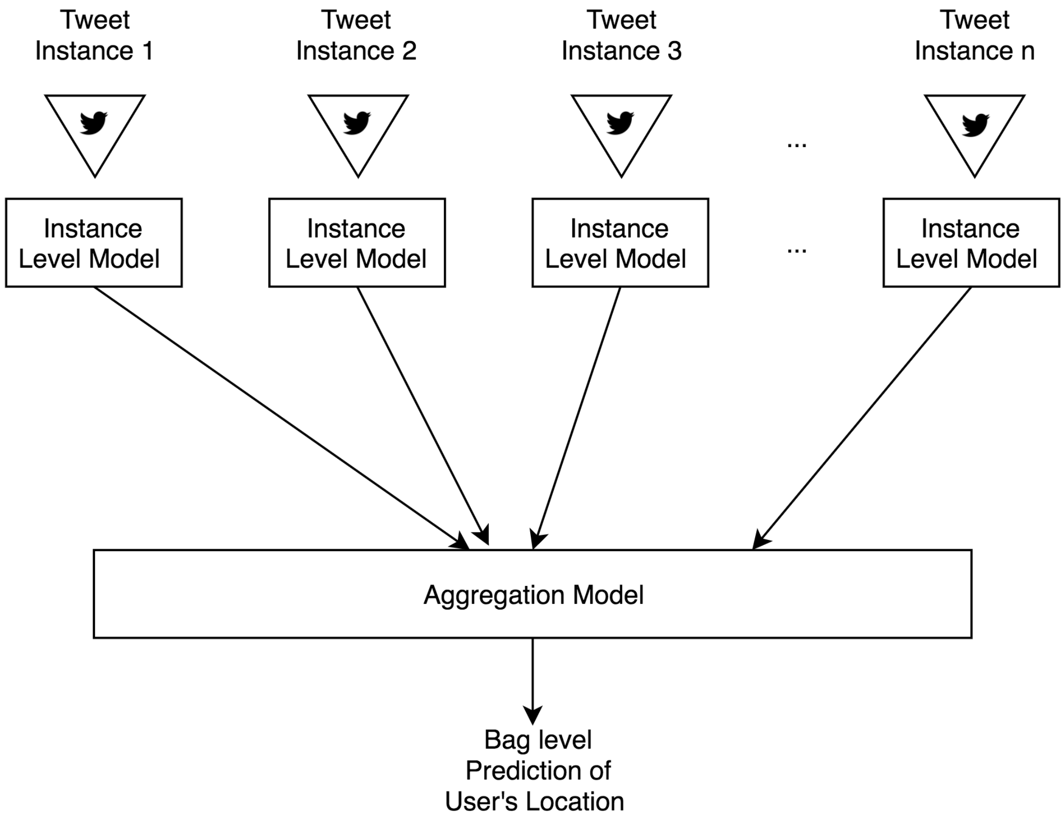

Figure 3.2: Model Architecture - This general figure provides the framework for multiple instance learning with neural network where instance level models feed their predictions to a bag level aggregation layer and losses are subsequently back-propagated through the network to learn the weights. In effect, all instance level models share weights and are actually a single model being learned.

away from a rigid logic based paradigm.

As another point of improvement over kernel based methods prevalent in prior research, milNN learns distributed representations which are implicit to a similarity measure in this case. Moreover, it doesn't require high compute and memory restrictions and can be learned using stochastic gradient descent which is easy to parallelize.

Additionally, as described in the problem formulation there are no assumptions on the membership proportions within the bag. The aggregation assumption of computer the average is also not particularly stringent as the sigmoid layer doesn't provide exact labels

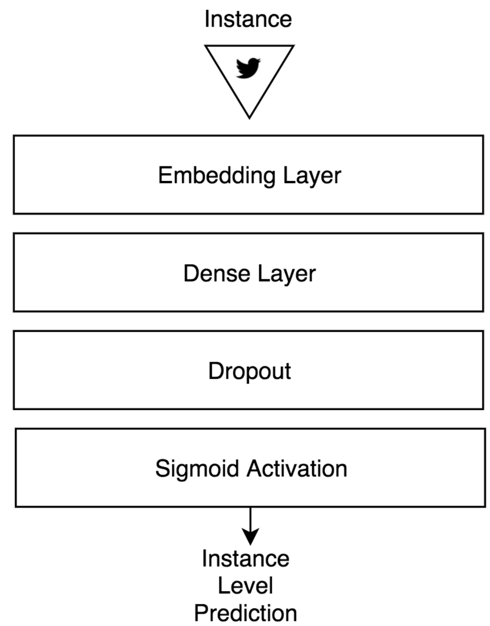

Figure 3.3: Instance Level Model - Consists of an embedding layer that learns a vector for each word in the tweet which feeds into a fully connected layer before being classified as belonging to the city or not. Dropout is added to regularize before the sigmoid activation.

for the instances, but rather a probability which is averages across all tweet classifications for a user level label.

The architecture is also flexible and the model described here can be seen as one example of the possibilities. Figure 3.2 describes a high level abstraction of the general idea of weights being shared in a wide sense and all the constituent parts can be replaced to accommodate the needs of the data.

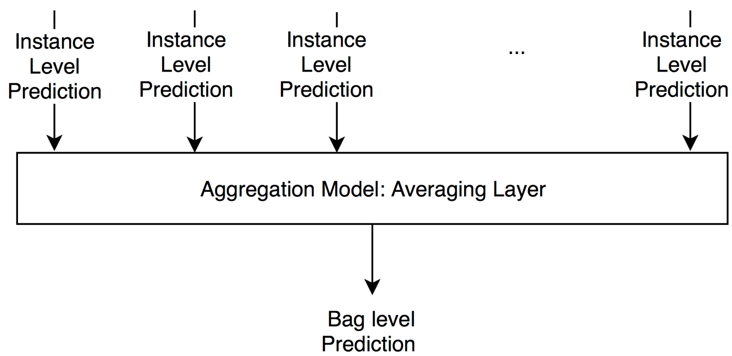

Figure 3.4: Aggregation Model - This model takes instance level predictions and averages them arithmetically to provide the bag level prediction which can be compared to labels and provide a loss that can be back-propagated through the network.

Chapter 4: Results and Evaluation

The following sections describe the dataset, the experimental setup with the hyperparameters of the model and training details. It then goes on to describe the Accuracy, F-measure and Running time comparisons of the various MIL architectures on the datasets to contrast with milNN. Finally, illustrative examples of the milNNs performance are shown as relating to the most indicative datasets.

4.1 Datasets

The datasets were created by reverse geocoding lat/long information from Twitter North America dataset[49]. These latitude and longitude readings were recorded when the user signed up location and subsequently tweets were recorded for this user. For use in this work, the top cities in the data were split into 15 datasets of equal number of positive and negative samples from the reverse geocoded dataset[49]. The negative samples for each city were randomly selected from the rest of the dataset after stratified sampling from other cities. Table 4.1 describes the number of training and testing instances in the respective datasets. The New York City data is the biggest with 19000 total samples. Chicago, San Francisco, Philadelphia, Washington DC and Toronto have more than 5000 examples and hence the results related to these datasets carry more weight. The rest of the datasets also have at least 2100 total users.

It is evident that there are some differentiating relationships that exist at the bag level at least by observing the results of the Multinomial Naive Bayes Classifier which averages at 70.3% accuracy. The highest accuracies being achieved on San Francisco, New York City and Philadelphia out of the bigger datasets.

It is also interesting to observe the results of a Support Vector Classifier on the dataset

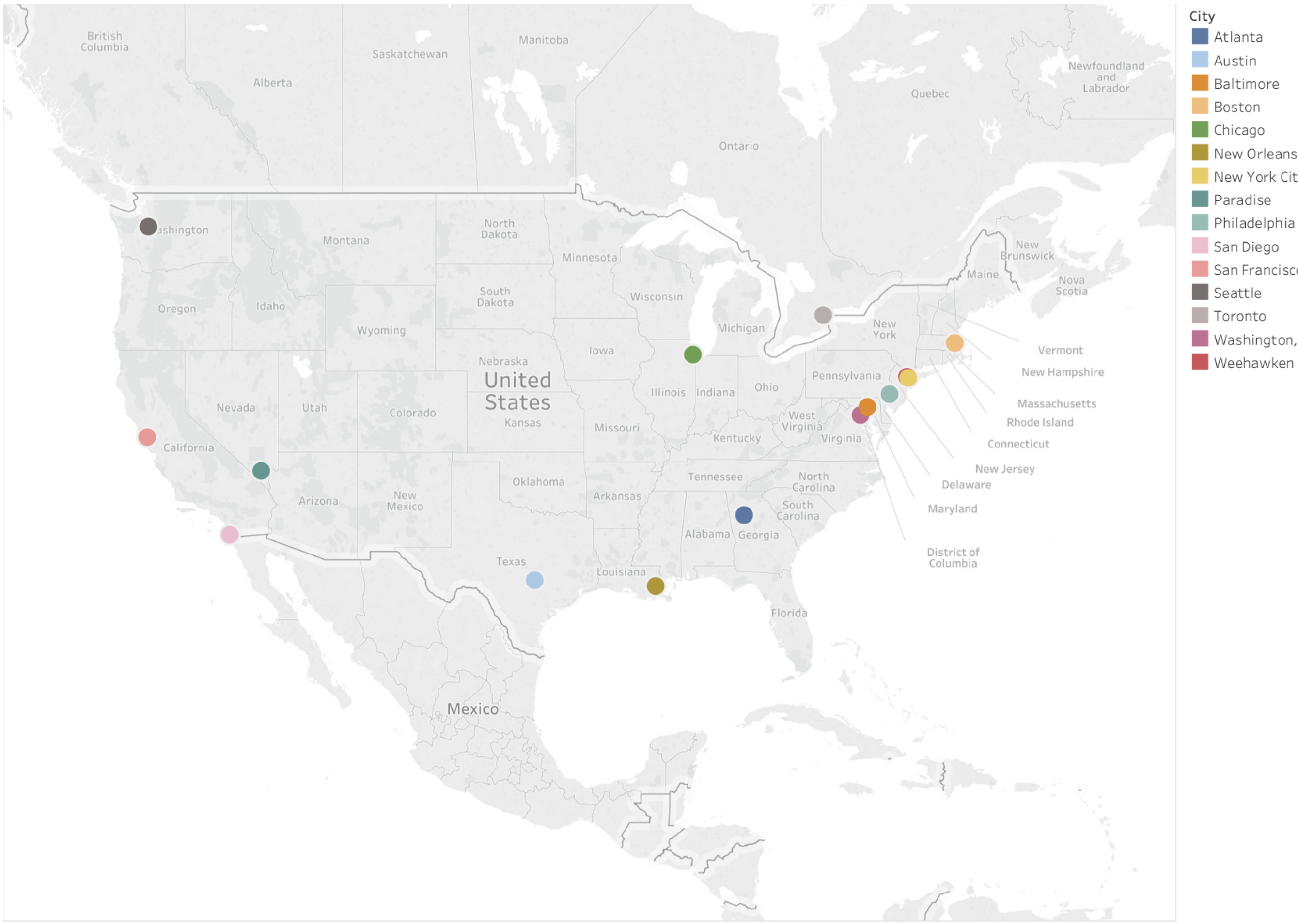

Figure 4.1: Datasets map - This figure indicates the cities under consideration in this work. A color legend is included which will remain constant throughout this work for indicating cities in figures that follow.

to see preliminary existence of nonlinear relationships that might be present in the dataset. San Francisco and New York datasets perform better on this initial litmus test of non linearity in language use. However, the Philadelphia data has a borderline classification accuracy of about 50% using the Support vector classifier.

Table 4.1: Datasets - Among the bigger datasets are NYC and SF. Cities like Boston and San Diego rank lower on the amount of tweets found in the dataset. It should be noted that only half of the total are positive examples. The negative examples were compiled randomly after stratified sampling from other cities.

City	Train	Test	Total
Atlanta,GA	3531	883	4414
Austin,TX	2332	583	2915
Baltimore,MD	2159	541	2700
Boston,MA	1911	478	2389
Chicago,IL	6628	1658	8286
New Orleans,LA	2072	520	2592
New York City,NY	15200	3800	19000
Paradise,NV	2475	620	3095
Philadelphia,PA	4633	1159	5792
San Diego,CA	1960	492	2452
San Francisco,CA	6168	1542	7710
Seattle,WA	2680	670	3350
Toronto,Canada	4029	1008	5037
Washington,D.C.	4584	1148	5732
Weehawken,NJ	1756	440	2196

4.2 Experimental Setup

Instance level model: The tweets are preprocessed by changing URLs, @mentions, and hashtags to a generic word for each. This choice was tested on datasets and while the accuracy was comparable, the instance level results were drastically different. For instance the model for San Francisco predicted 0.99633 for the words "San Francisco" under the chosen model and 0.06842 under the model containing hashtags and mentions. Thus, this preprocessing method was chosen so as to isolate language features from graph features of the dataset. Subsequently the tweets are tokenized and the top 5000 vocabulary words are henceforth considered in the modelling process. The tweet is then padded to a 20 word maximum (16 being the average for a tweet with 140 characters) and then fed through an embedding layer with 32 dimensions which is randomly initialized. Following this, there is a single hidden layer with 100 nodes that process the various language level relationships and feed the relu activations to the sigmoid layer for classification after adding a dropout

Table 4.2: Initial high level results on datasets: As a way to explore the capabilities of the data, basic bag of words classifiers are explored to demonstrate existence of features within the dataset. Multinomial Naive Bayes picks up frequency based capabilities, while Support Vector Classifier tries to model non-linearities at the user level. While these methods show high performance in some cases, they do not train any instance level classifiers.

City	MNB	SVC
Atlanta	0.7022	0.5005
Austin	0.7238	0.5026
Baltimore	0.7338	0.5009
Boston	0.6276	0.5000
Chicago	0.6701	0.5723
New Orleans	0.6731	0.5981
New York City	0.7076	0.5795
Paradise	0.6903	0.5016
Philadelphia	0.7092	0.5038
San Diego	0.6626	0.5671
San Francisco	0.7607	0.5797
Seattle	0.7388	0.6104
Toronto	0.7639	0.5009
Washington,D.C.	0.6760	0.5400
Weehawken	0.7114	0.5636

of 25% for regularization.

Bag level model: N is set to 10 and the outputs of the instance level models are averaged at a higher layer for the bag level output. At this level binary cross entropy loss is calculated using the bag level labels and back-propagated using the Adam optimizer. A batch size of 256 bags at a time is chosen and trained for 200 epochs with a learning rate of 0.0001. An early stopping condition is included which breaks out of training when the loss of the epoch converges and waits for 5 iterations to confirm the convergence.

4.3 Exploring the Hyper-parameter space

As the hyper-parameter space in neural networks can be large, here we only consider the implications changing of embedding dimensions and number of nodes in the dense layer of the neural network. The batch size, epochs, stopping conditions, inputs, optimizers and general architecture remain the same through these tests.

For exploring the model architecture, choices for embedding space and number of dense nodes was considered as shown in figure 4.2. While these choices reacted differently on the 15 datasets, a general conclusion to be made was that the accuracy wasn't particularly sensitive to these choices. However, the running time[1] was affected as the complexity in the model grew. As expected, the simplest model runs the fastest. However, a trade-off needs to be considered between adding complexity in terms of dense layer nodes (and/or embedding dimensions) and running time add on. As per figure 4.2 it was particularly fruitful to move from 10 embedding dimensions to 32. However, not so much from 32 to 50. Additionally, the dense layer nodes seem to perform close to each other, perhaps due to the regularization mechanism of dropout applied in the model which essentially reduces all these models to an ensemble[50].

It is also useful to see the models in the context of older models and this figure provides the scale at which the neural network models perform vs the rest by adjusting the axes to future comparisons.

[1]MacBook Pro (2016)- Processor: 2.9 GHz Intel Core i7 ; Memory: 16 GB 2133 MHz LPDDR3

Table 4.3: Hyper-Parameter Accuracy(dense nodes/embedding dimension): This table makes a case of higher complexity as the accuracy scores are bolder towards bigger embeddings and more nodes in the dense layers

City	50/32	100/10	100/32	100/50	150/32	200/32
Atlanta	0.6704	0.6648	0.6659	**0.6716**	0.6546	0.6478
Austin	0.6981	0.6895	0.6895	**0.7084**	0.7050	0.6964
Baltimore	0.6802	0.6784	**0.6932**	0.6802	0.6839	0.6839
Boston	0.6381	0.6088	0.6276	**0.6360**	0.6130	0.6255
Chicago	0.6574	0.6598	0.6538	**0.6616**	0.6520	0.6592
New Orleans	0.6731	0.6712	0.6865	0.6846	**0.6885**	0.6846
New York City	0.7005	**0.7061**	0.6995	0.6995	0.6971	0.6989
Paradise	0.6661	0.6500	0.6581	0.6500	0.6645	**0.6710**
Philadelphia	**0.6739**	0.6644	0.6713	0.6704	0.6687	0.6626
San Diego	0.6728	0.6768	0.6829	0.6768	0.6931	**0.6951**
San Francisco	0.7490	0.7549	0.7484	0.7549	**0.7562**	0.7484
Seattle	0.7254	0.7254	0.7239	0.7373	**0.7358**	0.7343
Toronto	0.7579	0.7579	**0.7629**	0.7560	0.7579	0.7569
Washington,D.C.	0.6446	0.6481	0.6446	0.6524	**0.6533**	0.6490
Weehawken	0.6841	0.6886	0.7068	0.6864	**0.7136**	0.7136

Thus, in order to be able to model complexity, the embedding dimension was chosen to be 32, and 100 dense nodes were picked as the model seemed to have considerable slowing down that couldn't compensate with improvement in accuracy when moved to the choices of 50 and 200 respectively.[2]

[2]Minor differences in the numbers in the next section stem due to inclusion of the validation set in training

Table 4.4: Hyper-Parameter F-score(dense nodes/embedding dimension): Even though the model is not particularly sensitive to the hyper-parameter changes here(all f-scores are within a few decimal points of each other), the higher f-scores occur towards the right of the table where the model is more complex

City	50/32	100/10	100/32	100/50	150/32	200/32
Atlanta	**0.6704**	0.6590	0.6621	0.6620	0.6530	0.6454
Austin	0.6879	0.6785	0.6750	**0.6986**	0.6884	0.6822
Baltimore	0.6754	0.6615	**0.6844**	0.6814	0.6755	0.6827
Boston	0.6214	0.5944	0.6164	**0.6217**	0.5843	0.6049
Chicago	0.6405	0.6385	0.6408	0.6447	0.6196	**0.6480**
New Orleans	0.6852	0.6919	**0.7031**	0.6952	0.6811	0.6906
New York City	0.6908	**0.7022**	0.6990	0.6984	0.6991	0.6946
Paradise	0.6521	0.6226	0.6467	0.6353	0.6426	**0.6542**
Philadelphia	**0.6942**	0.6819	0.6890	0.6774	0.6923	0.6755
San Diego	0.6414	0.6505	0.6579	0.6475	0.6725	**0.6739**
San Francisco	0.7527	0.7542	0.7529	0.7598	**0.7599**	0.7487
Seattle	0.7237	0.7212	0.7201	0.7349	**0.7346**	0.7311
Toronto	0.7549	**0.7550**	0.7539	0.7505	0.7530	0.7482
Washington,D.C.	0.6099	0.6300	0.6215	**0.6323**	0.6151	0.6411
Weehawken	0.6667	0.6746	0.6921	0.6584	0.6897	**0.6942**

Table 4.5: Hyper-Parameter Running Time(dense nodes/embedding dimension) : The simple model runs the fastest.

City	50/32	100/10	100/32	100/50	150/32	200/32
Atlanta	28.21	**20.64**	30.05	36.11	33.05	36.54
Austin	21.18	**14.35**	28.45	38.61	32.51	36.14
Baltimore	22.34	**13.38**	26.35	36.30	29.67	30.41
Boston	19.42	**11.59**	23.35	34.69	26.66	26.96
Chicago	40.44	**31.40**	48.98	49.67	42.32	46.82
New Orleans	20.32	**13.05**	25.43	37.31	29.00	32.28
New York City	71.13	**40.52**	74.34	90.54	72.27	77.30
Paradise	22.61	**14.50**	27.61	33.18	30.66	26.79
Philadelphia	39.17	**26.37**	40.38	46.61	48.86	50.51
San Diego	18.77	**11.41**	23.77	33.13	27.50	31.09
San Francisco	52.52	**34.02**	49.25	66.55	55.96	59.95
Seattle	26.50	**15.60**	32.33	42.74	32.29	37.71
Toronto	38.16	**24.58**	39.29	46.23	46.70	55.86
Washington,D.C.	31.91	**26.15**	32.84	41.65	33.85	42.26
Weehawken	16.89	**10.43**	21.23	28.47	24.39	24.27

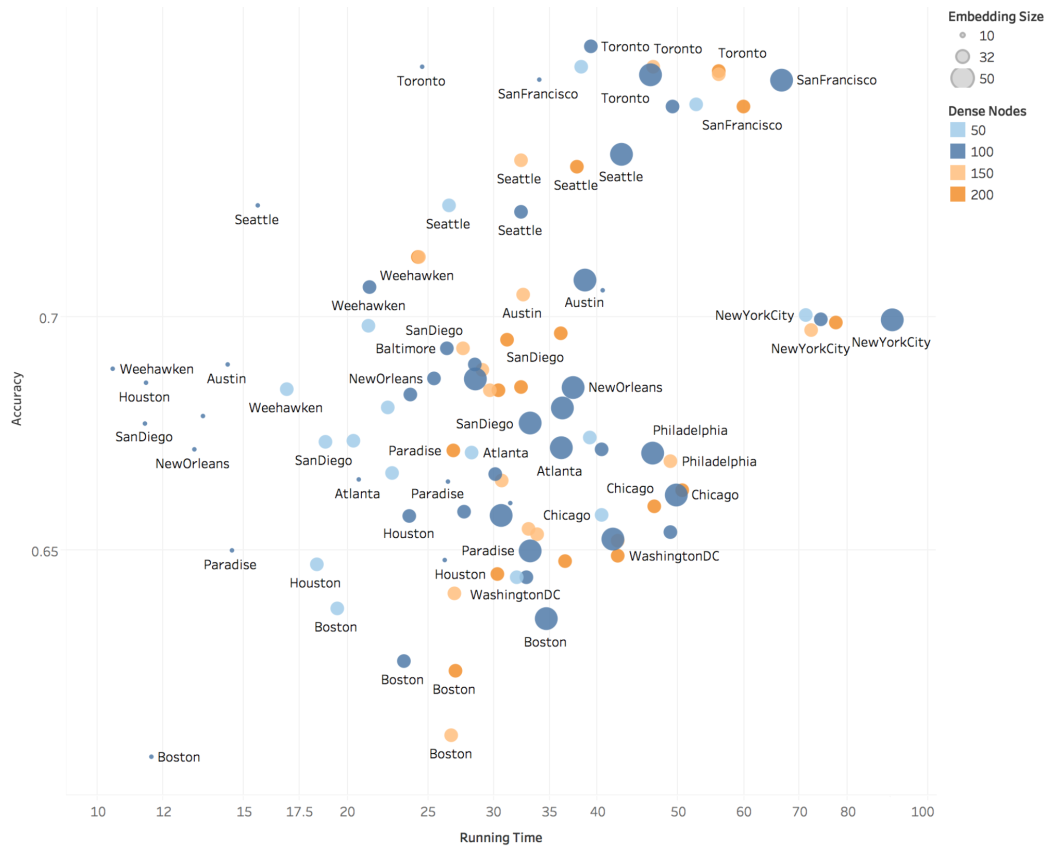

Figure 4.2: Hyperparameter Comparison - In the above visual, size is the embedding size, color is number of nodes in the dense layer, cities are labeled near the datasets. The running time is on the x-axis and accuracy is on the y-axis. Here the upper left corner of the graph is better where the accuracy is high and running time is low. Another characteristic that is observable is how the results of a particular model might cluster together to demonstrate the robustness of the model given various different datasets.On this count the lower embedding dimension (faster) models don't impress as accuracy remains low on many datasets and results seem more scattered.While the higher embedding dimensions remedy the accuracy, they increase the running time. It is however, useful to notice that the difference in accuracy is insignificant compared to running times as the models get more complex.

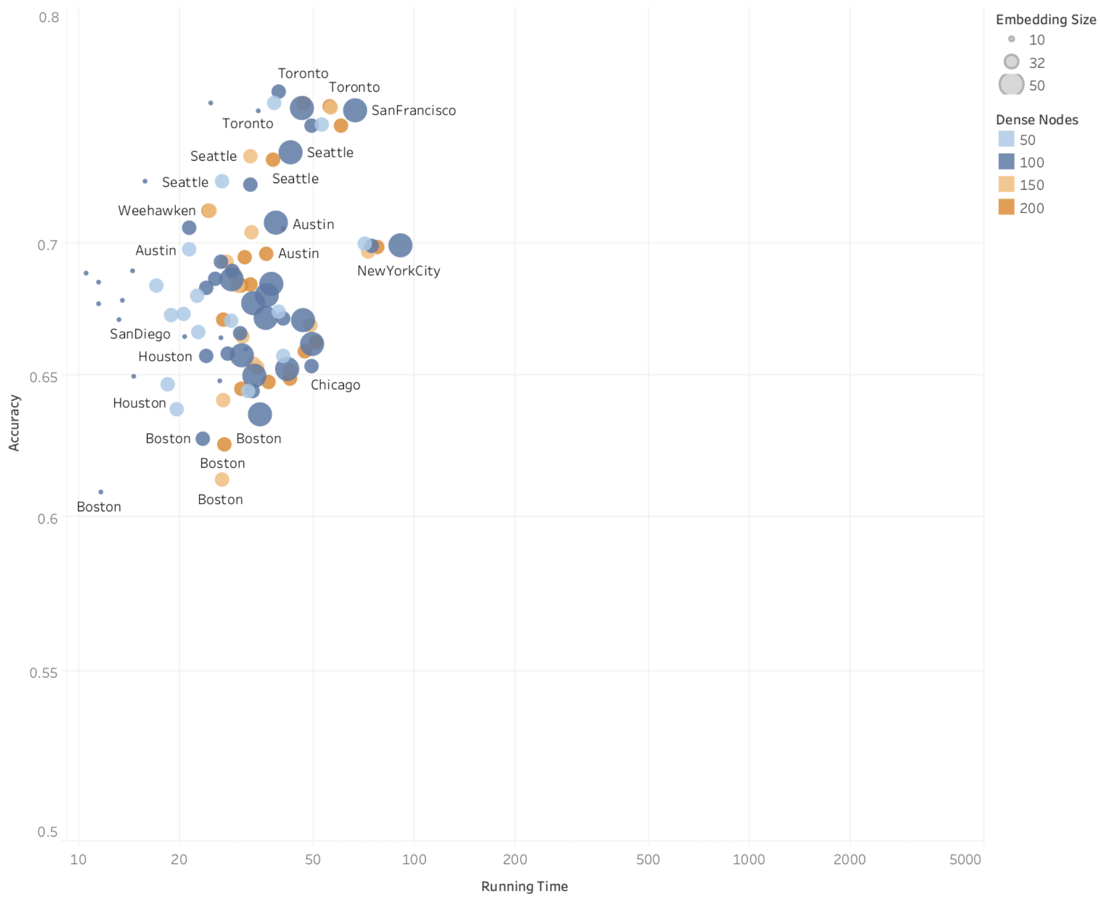

Figure 4.3: Context of proposed model versions - This scales the graph to prepare the reader for the context about to be presented in the next section where the proposed model is compared with prior work. This suggests that even at its worst settings the model would be able to demonstrate superior accuracy/runtime trade-offs.

4.4 Performance comparison with MIL methods

To judge the performance of milNN, it is compared with prior work in MIL namely SIL, MISVM and GICF . Here, the metrics of interest - Accuracy and F measure are compared and the running time comparison is given detailed treatment in the next section with this context.

As per the results in Table 4.2 the classifier has a higher accuracy on the bigger San Francisco and New York datasets. It outperforms all other methods on 14 of the 15 datasets considered. For the exception, the Paradise, NV dataset is one of the smaller ones and does better with the GICF approach. However, milNN outperforms GICF in the F score calculation of results (Table 4.3).

milNN outperforms all other methods on 10 out of the 15 datasets under consideration for the F measure. It loses to SIL on the Paradise and Washington DC datasets. Of these, the DC dataset is bigger and on observation the proposed model comes within the third decimal point while using a fraction of the time and still outperforms GICF by a considerable margin, as it does for the Paradise data. The method loses to GICF on the Atlanta, Baltimore and Philadelphia datasets; however, GICF takes longer to run and can't perform as well on the accuracy metric either. Additionally, it is worth noting that F scores are not as consequential as accuracy given the balanced nature of the dataset.

Table 4.6: Accuracy - milNN performs higher 14 of the 15 datasets under consideration.

City	MISVM	SIL	GICF	milNN
Atlanta	0.4990	0.5780	0.6025	**0.6602**
Austin	0.4970	0.6070	0.6501	**0.7015**
Baltimore	0.4990	0.5180	0.6248	**0.6858**
Boston	0.5000	0.5460	0.5774	**0.6276**
Chicago	0.5000	0.5760	0.6429	**0.6502**
New Orleans	0.5000	0.5230	0.6365	**0.6962**
New York City	0.5000	0.5740	0.6476	**0.7024**
Paradise	0.4980	0.6060	**0.6629**	0.6565
Philadelphia	0.4960	0.5190	0.6195	**0.6644**
San Diego	0.5000	0.6300	0.6504	**0.6850**
San Francisco	0.5000	0.6300	0.7322	**0.7542**
Seattle	0.5000	0.6210	0.6970	**0.7269**
Toronto	0.4990	0.6390	0.6895	**0.7520**
Washington,D.C.	0.5000	0.5740	0.6298	**0.6437**
Weehawken	0.5000	0.6160	0.6727	**0.7000**

Table 4.7: F measure - The proposed model outperforms on 10 of the 15 datasets under consideration.

City	MISVM	SIL	GICF	milNN
Atlanta	0.0000	0.6900	**0.6982**	0.6568
Austin	0.0000	0.6650	0.6291	**0.6848**
Baltimore	0.0000	0.6560	**0.6957**	0.6816
Boston	0.0000	0.5820	0.5960	**0.6130**
Chicago	0.0000	0.5180	0.5163	**0.6420**
New Orleans	0.0000	0.6690	0.6976	**0.7041**
New York City	0.0000	0.6230	0.6349	**0.6988**
Paradise	0.0000	**0.6510**	0.6365	0.6468
Philadelphia	0.0000	0.6620	**0.7006**	0.6830
San Diego	0.0000	0.6080	0.6325	**0.6548**
San Francisco	0.0000	0.6980	0.7398	**0.7603**
Seattle	0.0000	0.6840	0.7112	**0.7215**
Toronto	0.0000	0.6810	0.7056	**0.7485**
Washington,D.C.	0.0000	**0.6190**	0.4910	0.6108
Weehawken	0.0000	0.6590	0.6588	**0.6827**

4.5 Running Time Comparison

milNNs running time comparison provided a 15 out of 15 improvement in comparison to the other methods. This underlines the guaranteed contribution of this work in terms of speed and optimization. Another place the proposed model saves time is in its negation of feature engineering and minimal preprocessing. However, these times can be quite arbitrary based on what feature vector the domain experts (linguists in the current context) decide to come up with and have not been documented here.

The GICF and milNN experiments were run on a consumer laptop[3] and the MISVM and SIL algorithms required a cluster with higher memory requirements[4]. Thus, the first two methods have running times that are the upper bounds and would be considerably lower on faster machines used on MISVM and SIL. Conversely, the MISVM and SIL running times can be thought of as being at least this high on consumer computers.

The computational complexity could be observed in action for MISVM and SIL running into hours of running on sampled instances. MISVM, even though theoretically superior for MIL than SIL, learns nothing from the hours of training which is reflected in the 50% accuracy across the datasets and zero F-score despite the longest running times. Perhaps this can be attributed to the strong membership assumptions and OR aggregation function which doesn't apply to the current use case. Due to these results, MISVM is not considered any more comparisons henceforth.

SIL does considerably well as the simplicity generalises to the user tweet relationship of the dataset for geolocation. Even though, SIL finishes training faster than MISVM, the method is still considerably slower that the neural network methods (GICF and milNN). The method proposed in this paper, milNN, provides an average 450x speedup over MISVM, 70x speedup over SIL, and 4x speedup over GICF.

[3] MacBook Pro (2016)- Processor: 2.9 GHz Intel Core i7 ; Memory: 16 GB 2133 MHz LPDDR3

[4] Argo Research Cluster - Processor: 64 Core AMD Opteron; Memory: 512 GB(50 used)

Table 4.8: Running Time - Consistently and significantly better performance is demonstrated by the proposed method (milNN)

City	MISVM	SIL	GICF	milNN
Atlanta	38,799	3,031	106	**38**
Austin	21,304	3,072	81	**34**
Baltimore	27,069	3,030	69	**35**
Boston	19,166	2,724	60	**28**
Chicago	22,935	2,866	288	**45**
New Orleans	33,043	2,826	76	**52**
New York City	10,096	3,448	822	**85**
Paradise	12,891	2,825	149	**40**
Philadelphia	14,453	3,320	212	**52**
San Diego	22,547	3,108	67	**57**
San Francisco	12,562	3,883	222	**64**
Seattle	17,088	3,738	143	**41**
Toronto	17,861	3,847	131	**48**
Washington,D.C.	21,026	2,790	193	**42**
Weehawken	9,460	1,894	63	**29**

4.5.1 Running time vs Accuracy

While comparing running times to accuracy, in Figures 4.4 and 4.5, it is evident that milNN outperforms all other methods. Even on the datasets where accuracy was lower than others(Paradise), it is noticeable that the difference in accuracy is not as significant and the speed up provided by milNN.Thus the gain in performance outweighs the insignificant loss in accuracy for this particular dataset.

In Figure 4.5, it can be seen that the upper left corner of the graph is the more optimized area where accuracy is high and running times are low. Based on this, SIL is the worst performer as running time is high and accuracy is lowest as it occupies the lower right hand corner of the graph.

Another interesting observation to be made in Figure 4.5 is that milNNs accuracy vs running time scores are clustered together suggesting consistency and robustness of the approach when faced with various datasets with different feature sets. Conversely, GICF and SIL demonstrate a spread out scatter plot, indicating inconsistent performance across datasets.

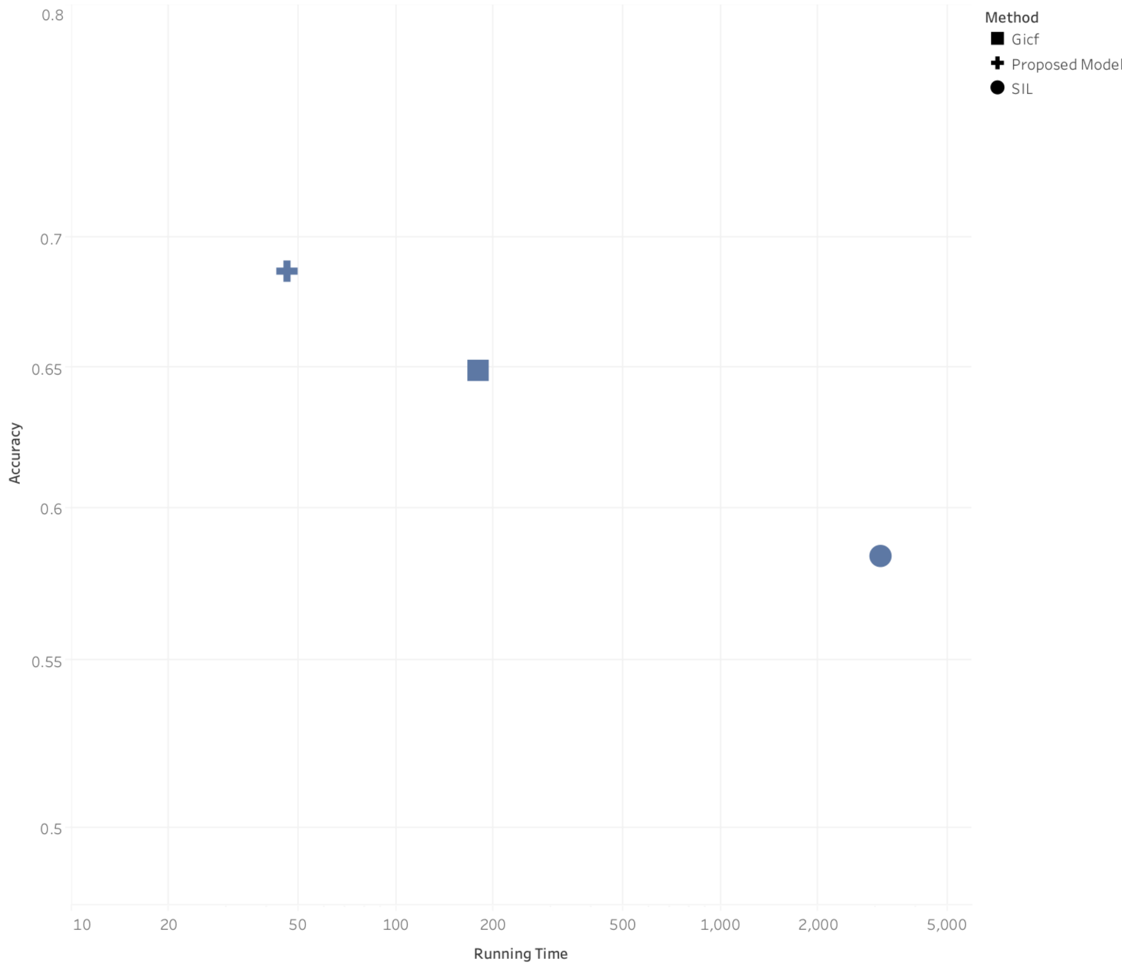

Figure 4.4: Accuracy vs Running Time - milNN has the highest average accuracy in the lowest average running time.

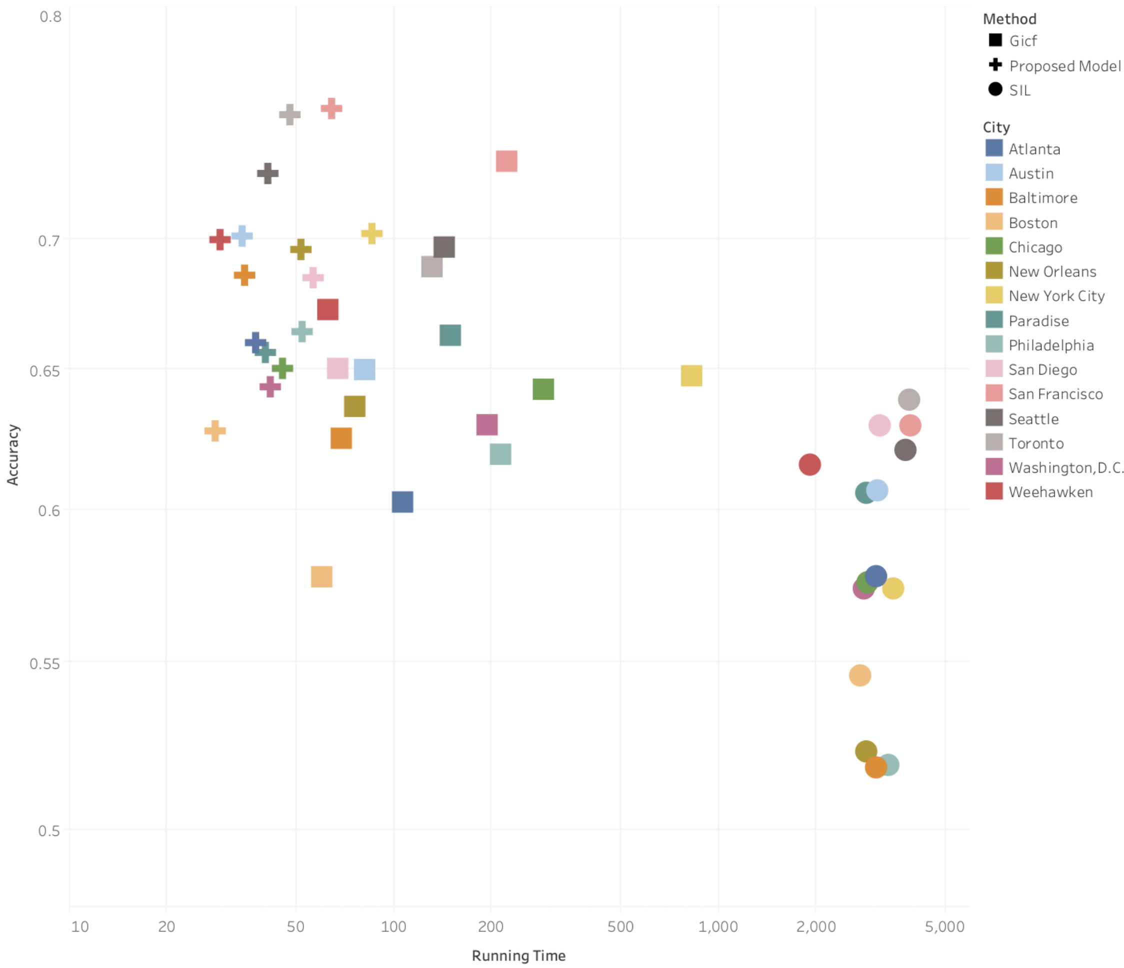

Figure 4.5: Accuracy vs Running Time by City - The results of the proposed model (+) plus signs are consistently faster and better while older methods have bigger spread in the results with SIL (•) round bullets being the slowest and most inconsistent and GICF(■) squares being somewhat faster but less reliable given diverse requirements of datasets.

4.5.2 Running time vs F measure

Figures 4.2 and 4.3, indicate graphs that are particularly interesting as the method couldn't outperform every old method on this metric.

SIL continues to occupy the right hand side of the graph due to higher running times. GICF remains modestly in the middle. However, they are both more scattered in terms of results with values below 0.6. This effect can be observed most predominantly the Chicago and Washington DC datasets. This questions the variability in these methods which is most probably an effect of feature engineering requirements of different datasets.

milNN clearly wins as the results for all datasets remain clustered in the top left hand corner of the graph which is the best quadrant to be in with high F scores and low running times. All F scores are also greater than 0.6 and the scatterplot is fairly clustered together which speaks to the robustness of the method.

Thus, after the comparison were made by extensive experiments on 15 datasets, the proposed model demonstrated superior scalability and running time performance while also being robust to the changing needs of various datasets.

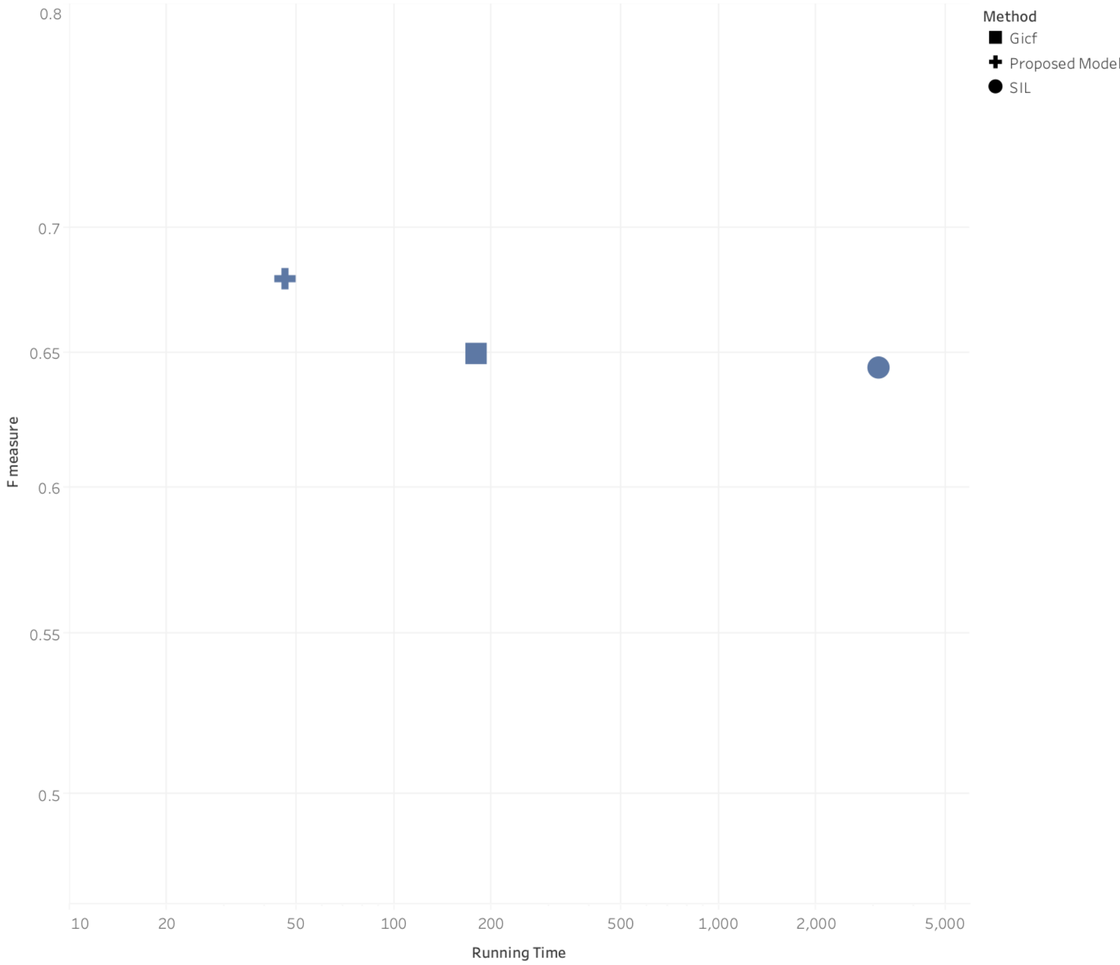

Figure 4.6: F measure vs Running Time - The proposed model performs with the highest average f score in the shortest average running time over all datasets while the round bullets (SIL) and square (GICF) are slower and not as clustered together as the + signs of milNN, the proposed model.

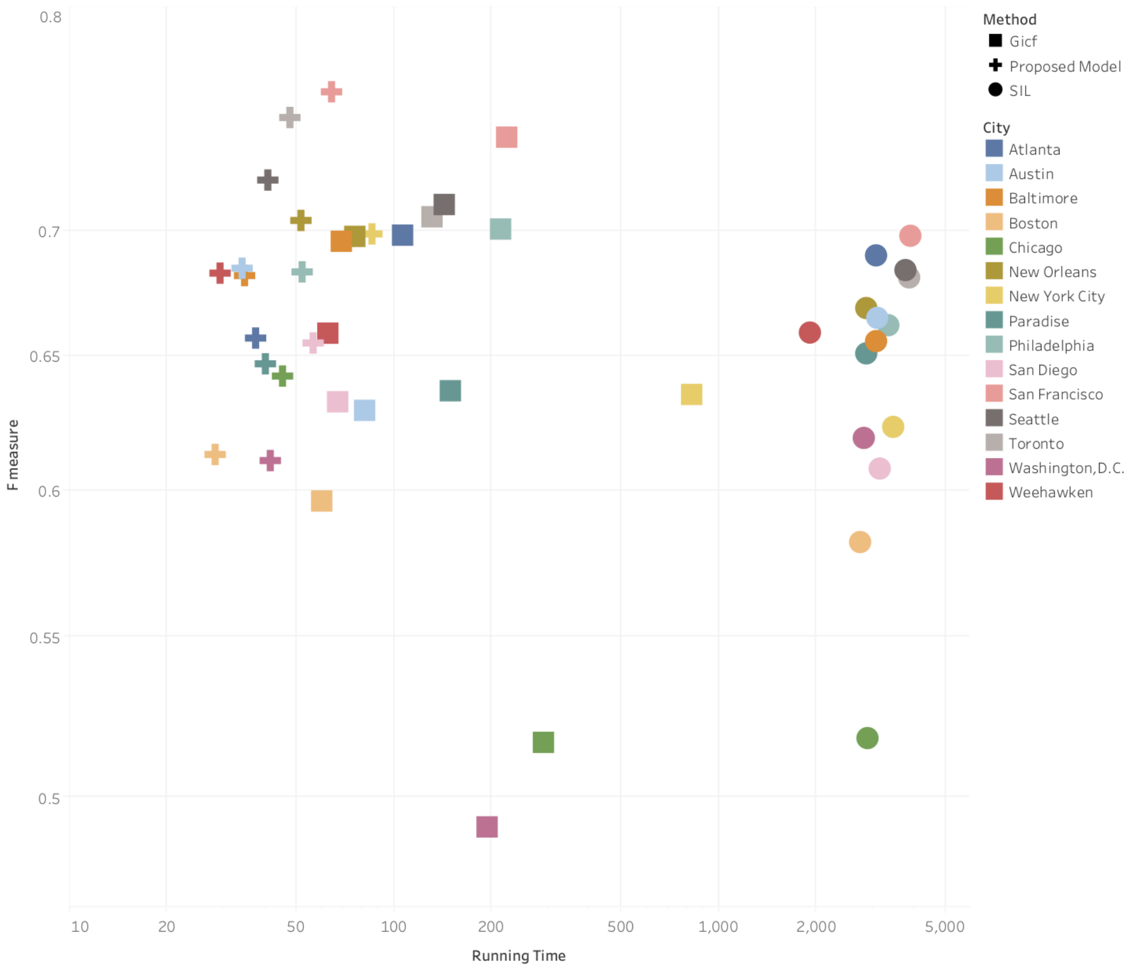

Figure 4.7: F measure vs Running Time by City - milNN's results are consistently high even in diverse datasets and the + signs cluster in the best area of the graph where running times are low and metric is high.

4.6 Illustrative Examples

Figure 4.8: Word Cloud for NYC - It can be seen that the the word cloud created has a range of location names indicating the NYC users' proclivity to tweet their location.

The following sections deal with the biggest and highest performing language geolocation datasets - New York City and San Francisco.

The word cloud visual is created using all the test instance tweets that were classified to be over 0.95 by the instance level classifier. Location entities were subsequently extracted from these tweets using StandfordNER [51] and then weighted into a word cloud. High level comparison of the SF word cloud to the NYC word cloud indicates that NYC tweets mention various locations throughout the city more than SF tweets do. At the very least, the models learned by these datasets tend to pick out more place name features for NYC than SF. Hence, one would also expect there to be more language level features to be discovered in the SF dataset given the high accuracy of the method on these tweets.

The structure of the following analysis is to go over examples of users that belong

Figure 4.9: Word Cloud for SF - In contrast, the San Francisco word profile doesn't consist of as many place names as New York City and the model was able to determine higher level language features from this data.

to the city to analyse language patterns discovered by the model. Additionally the anti-patterns are explored by analysing tweets from users that do not belong to New York and SF respectively in order to discover how these cities don't tweet. The tweets being classified as mostly certain are colored green (prediction over 0.95), the undecided tweets are colored orange (prediction between 0.4 and 0.6) and those that the models found definitely not belonging to the particular city are colored red (prediction below 0.4). They will be referred to as green, orange and red tweets respectively in the following analysis

4.6.1 New York City

User 1 We can clearly see that the model can figure out place names by itself that relate to a particular location. It found Brooklyn, NYC, New York. In the second case, it actually classified the tweet certainly as from New York because of multiple mentions of the name of the city.

Another indicative example can be seen in the mention of bags being lost by United Airlines. It is noteworthy that during the preprocessing phase, the hashtags and @mentions were abstracted away from the model. This might be indicative of people thanking the airlines for losing their baggage nationwide. Thus the model is not quite certain where the tweet is from.

In this classic case where the bag level label doesn't hold, the user is traveling to Dallas and Chicago. The model caught this fact even though it did not have the label at the tweet level.

User 2 This example is particularly indicative of someone in New York. The model clearly catches the easy location names like Lincoln Tunnel and New York City with near certainty. The more interesting nuances can be noticed in the tweets when the "view is beautiful" (high probability) vs when "the beautiful view overlooks the ocean" (low probability). The model realizes that even though NYC has views they don't overlook the ocean. Another interesting thing to note about this ocean tweet is the mention of the Hilton hotel. In the undecided

New York City - User 1

1.0 - " I'm at Fashion's Night Out NYC (NYC, New York) w/ 158 others http://t.co/r2XDXEp "

0.99496824 - " I'm at Brooklyn Bridge w/ 2 others [pic]: http://t.co/ry1BAJZo "

0.43607074 - " Piece of crap airline! Thanks for losing my bag! @united @UnitedAirlines #united #unitedairlines "

0.014939007 - "I'm at Sfuzzi (2533 Mckinney Ave, Routh St, Dallas) w/ 7 others http://t.co/BnxbYtSr "

0.0078560486 - " I'm at Public House (400 N State St, at Kinzie, Chicago) w/ 7 others http://t.co/q9tiqW2m "

Figure 4.10: New York Tweets: User 1 - For demonstration of location related entities in high and low probability tweets.

orange tweet we can see how pervasive Hilton hotels are and they might not necessarily be in any one place. However, with the mention of the ocean, the model decreases the probability that the tweet is from New York.

Another cultural indication is seen in the last red tweet which signifies how the activity of standing in line at Walmart is not common to New Yorkers as there are no Walmart stores in city.

User 3 This is an interesting example as the user is not from New York City, but Arlington, NY. The example is particularly illustrative as the user is near New York City and frequently travels there. This is indicated in travel related tweets that are marked green by the model despite the bag level label being not New York.

The model remains undecided about the cinema tweet mentioning The Twilight Saga due to the movie being popular nationwide.

New York City - User 2

0.99559683 - "Beautiful view... good food... great music... romantic husband = perfect evening (@ Le Kaveka Restaurant & Bungalows) http://t.co/sE4EjSGV '

0.99972242 - " A lovely fall brunch with @Accarrino (@ Anthony David's w/ @accarrino) [pic]: http://t.co/4wzMBut6 "

0.99978334 - " I'm at Lincoln Tunnel (New York City) w/ 3 others http://t.co/EY5B7SB5"

0.49836314 - "I just became the mayor of Hilton Moorea: Toatea Crepes & Bar on @foursquare! http://t.co/jopZTuuE"

0.2133007 - "Beautiful breakfast overlooking the ocean (@ Hilton Moorea: Arii Vahine Restaurant) [pic]: http://t.co/277WG6QF"

0.11722157 - "Standing in line to return what we bought last night. Efficiency! Walmart is out of cash. Waiting 15 min for refund. http://t.co/VIXay1dv"

Figure 4.11: New York Tweets : User 2 - For demonstrating nuanced place name recognition with better context.

There are standard location related red examples that were captured at the end.

Not New York City - User 3 (Arlington,NY)

0.99989402 - "Back in New Yawk Citay (@ Grand Central Terminal w/ 28 others) http://t.co/2CBNSMtJ"

0.99894804 - "Back to Vassar on the 2:45 Metro North... Snow fall was pretty while it lasted (@ Grand Central Terminal) [pic]: http://t.co/MRE8JqMk"

0.50644404 - "@paradisetaylor and I on date night ï\x98\x8a (@ Regal Columbiana Grande Stadium 14 for The Twilight Saga: Breaking Dawn -...)http://t.co/54ywGeq9"

0.018137755 - "I just ousted @aashim_91 as the mayor of College Center - Vassar College on @foursquare! http://t.co/qSIjibJA"

0.3946189 - "I just became the mayor of Matthew's bean on @foursquare! http://t.co/jLE2je5g "

Figure 4.12: Negative Example in New York City: User 3 - This user is from Arlington, NY which is near the city and yet the model can distinguish when the user is tweeting from within the city.

4.6.2 San Francisco

San Francisco was the best performing model and the dataset was bigger indicating the popularity of Twitter in SF. Thus, there were many interesting observations to be made from the illustrative examples that went above and beyond catching direct location mentions in individual tweets. The following examples illustrate, besides the obvious name place related classifications, a proclivity to stick to english grammar even while being restricted to 140 characters by the people in this city.

San Francisco - User 1

0.9999975- " I'm at Alcatraz (Alcatraz Island, San Francisco Bay, San Francisco) w/ 6 others http://t.co/47YWmX9p "

0.9999856 - " I'm at Chinatown Gate (500 Bush St, at Grant Ave, San Francisco) http://t.co/rb49RnFa "

0.5055542 - " @matthewharkin @phillo haha, now I'm worried ")

0.025480814 - " I'm at Tiffany & Co. (210 N Rodeo Dr., Beverly Hills) http://t.co/YppqS7ix ")

Figure 4.13: Positive Example in San Francisco : User 1 - Place name recognition related example for the SF instance level model

User 1 These are standard location names being caught by the model and are being rightly classified. The undecided tweet looks generic enough to not have any indicative pattern here. Hence, the model looks promising at this base level example.

User 2 However, sure enough, there is a mention of technology being labelled as green for SF by the model. This is a particularly indicative of an SF tweets considering the model did not have the explicit @mentions while prediction.

This users tweets also demonstrate the high level language features the model is capable

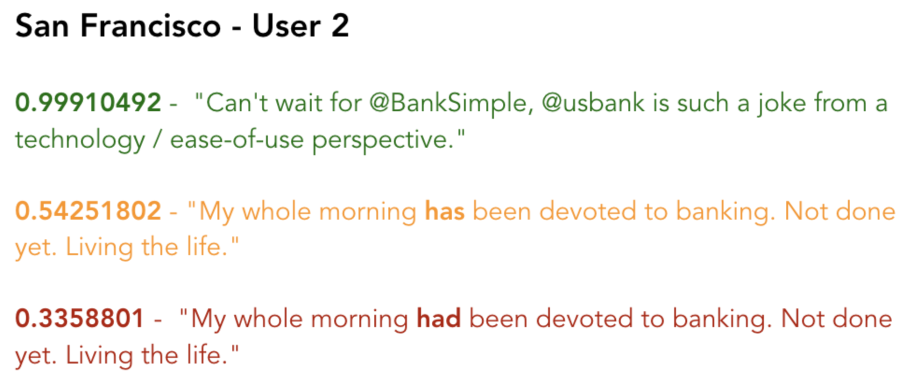

Figure 4.14: Positive Example in San Francisco : User 2 - Demonstration of higher language level features being learned by the model as it recognizes the better grammar choices in the undecided tweet and increases the probability of belonging to SF.

of dealing with thus filling the gap of all prior bag of words related geolocation research. The general tweet of wasting a morning at the bank is classified as though it could be from anywhere and the model remains undecided. However, it quickly lowers the probability of the tweet belonging to SF as soon as it notices the awkward grammar in the red instance which is only a one character difference from the orange tweet. No locations have been mentioned in either of these tweets. They don't contain any hashtags or mentions either.

User 3 The model would be remiss if it didn't capture the engineering and technology culture in the Bay area particularly as it relates to Apple Inc. This user's green tweets show that "Engineers love free food!" is certainly something that is indicative of San Francisco. The Mac and iOS mentions in the other two tweets also classify high. However, the really interesting tweet is the one that was classified red even though it mentions Apple by name. It is clear that the user is visiting the Cupertino campus of the company as there was a name tag provided. Hence, the model picked up on being away from San Francisco. This is an example of travel related discrepancies caught by the model without any place being mentioned.

San Francisco - User 3

0.97942388 - "Engineers love free food! #IDF2011 http://t.co/rThzEeKO http://t.co/oyYqSmYo"

0.99747145 - "@hashimwaheed the left one is USB serial into Mac, the other is normal iPhone USB into Mac"

0.9944582 - "iOS5 beta expires today! "limited-edition b7b" redsn0w lets you sync data+ pics: OSX http://t.co/EbVEGO0t Win http://t.co/vC5PK2Eg"

0.0067595979 - "@alexeheath my host at Apple surprised me with that visitor's name tag...I had expected it to be my real name :)"

Figure 4.15: Positive Example in San Francisco : User 3 - Technology related jargon is modeled high in SF social media as expected while qualifying these mentions with context.

User 4 This negative example is a clear indication of the SF population sticking to proper grammar while tweeting as every tweet by this person has low probability and not one of them has the correct sentence structure or spelling. The example that is particularly indicative of this is where the user mentions ”technical issues” in a badly formed sentence and still doesn’t receive any substantial probability for belonging to San Francisco.

Not San Francisco - User 4 (Louisiana,Arabi)

0.16304019 - "Follow the OG triple OG @thad4mayor to ensure that he don't steal ur wallet when he see you in the streets...<>jtfo"

0.23261635 - "@jbdachamp u show me no luv :("

0.0011654327 - "Nap time"

0.011714808 - "somethins gotta give"

0.10918618- "@Cree_Oh_Lay_CO how ya been?"

0.00020607341 -"@jbdachamp and u won't lol",

0.0094076423 - "@jbdachamp I was MIA 4 a min due 2 **technical** issues but now I'm baaaaack lol"

0.010172283 - "da best part is that the downs dont last always"

0.20761815 - "I luv fridays :)"

0.064976566 - "TGIF"

0.073392898 - "@Cree_Oh_Lay_CO we're great :)"

0.18137941 - "@thad4mayor "our" hmmm lol"

Figure 4.16: Negative Example in San Francisco :User 4 - This user belongs to Louisiana and has poor grammar choices in tweets. This is picked up by the model even when the word 'technical' is mentioned which should have rated higher for being from SF.

4.7 Insights from results

Thus, it was discovered after running extensive experiments on datasets from 15 locations that even at it's worst, the proposed architecture proved itself to run faster than the state of the art. It outperformed older methods on most datasets on various metrics and saved time by eliminating feature engineering and kernel design.

Subsequently, it demonstrated ability to capture different features in datasets that had considerably different needs with New York City tweets being location heavy and San Francisco Tweets adhering to strict grammar even within the 140 character limit. The same general model was able to cater to the needs of both datasets and successfully discovered patterns that were indicative of the local language style.

Chapter 5: Discussion and Conclusion

To help deal with the information deluge caused by content generation on social media platforms, this work focused on geo-locating tweets while working from user(set of tweets) level location labels. This was accomplished by providing a framework consisting of an end to end trainable neural network architecture for Multiple Instance Learning that helped transfer user level location labels to individual instances of tweets. The method was subsequently applied to datasets from 15 cities and was compared with state of the art approaches as well as various architecture designs. The model outperformed earlier methods and proved to be faster across all 15 datasets while eliminating the need for feature engineering by learning representations. This was subsequently demonstrated using illustrative examples where the model could identify place names and grammatical structure in language.

5.1 Limitations

Since the results heavily depend on the datasets, the models would need to retrain as the social media data formats change. For example, tweets have recently been increased from 140 character to 280. While this change can be accommodated by changing the words per tweet assumption during tokenizing tweets, it would require an updated dataset for training to catch the newer language patterns that might have emerged with the increased character limit. Many other changes have occurred with twitter data since the publication of the dataset in 2012[49] used to construct the demonstrations in this work, such as urls, photos, videos, quotes and @replies ceasing to be included in the 140 character limit[52].

Another major limitation of this work is the limit on the number of instances that can be considered while training being a constant since the multilayer perceptron is rigid in being able to only accept a fixed length input. In future work, this limitation could be remedied using nested recurrent neural networks which are capable of allowing tweets to be of any size and any number of tweets to be considered from a particular user.

An area of further improvement could stem from designing a robust instance level classifier metric. This work demonstrated the abilities of the model using illustrative examples, however a quantitative measure of the accuracy remains elusive as there is no way to test the model without labeled information at the instance level. Such labels might be intuitive where the use case is sentiment analysis [9], but they can be quite subjective for the geolocation modeling case as the architecture doesn't assume a particular characteristic associated with the cities under consideration. For example, if a subset of the data were to be labeled where locations are mentioned by name, it would only test the instance level model on being able to catch that particular pattern and would fail on instances where other characteristics may be present in the data like superior grammar usage. Another method for instance level classification judging is presented in [7], where instances are filtered based on sure classification a bag level model is evaluated. If the bag level model trains better with the new selected instances, then the instance level model is deemed to have discovered the underlying patterns in the instances. While this approach might suit a paragraph based dataset (news, reviews), it falls short for a collection of tweets as they share little context among each other in the bag. Additionally, the neural network is constructed to model non-linear relationships in the tweet level data which might not be caught at the bag level by a classifier. For example, if this model of evaluation was to be considered and a Mulitnomial Naive Bayes classifier was trained after triaging the tweets, instances that differ by a word would be considered to bring the same value even though one might be grammatically incorrect due to the wrong word placement as shown in Figure 4.14.

5.2 Future Work

As the architecture is modular and various pieces can be substituted to cater to particular dataset characteristics, many improvements can be made depending on the use case and instance level relationships in the datasets.

5.2.1 Transfer Learning

Any twitter related embeddings can serve as a starting point for the embedding based classifier[53][39]. It would then be equipped with some idea of what tweets are like and will retrain (or fine-tune) to serve the needs of the particular use case of geo-location.

The transfer learning approach will also generalise to other language based datasets with their particular targets like sentiment analysis.

5.2.2 Multiple Classification

Instead of 15 datasets, if a dataset with multiple locations were to be treated, a simple change from sigmoid to softmax layer classification will equip the model to deal with multiple locations at once. This approach would require some exploration as the model might be confused by similar place names in various cities(like 'Main Streets') or high level language patterns that are pervasive throughout the country. However, it might also be able to contrast language patterns better given exact labels for each instance as opposed to City X/not City X type labels.

5.2.3 Other instance level models

Depending on the complexity of the instance level data, a sequence model might be appropriate for classification given the success of Recurrent Neural Networks with sequential data[32]. This could be used to accommodate a variable number of instances per bag as the current Multi Layer Perceptron model is limited to accepting a fixed number of inputs.

In fact, the instance level classifier can be substituted by any trainable architecture including CNNs, ResNet blocks or other exotic architectures that might emerge, depending on the requirements of the dataset.

5.2.4 Exotic Aggregation Functions

While this work uses a basic average function to aggregate the instance level predictions to the bag level, this relationship might not be as straightforward for other use cases particularly if the instances are not relatively independent like tweets in each bag.

However, in this case it would be important to be careful so as to not interfere with the instance level model extracting information as a non linear relationship would reduce the instance level classifications to a spatial feature set generator.

5.2.5 Other Applications

Given the generality of the architecture, it can be leveraged to solve other Multiple Instance Learning problems in medicine and genetic research. The embeddings model as employed in milNN, would translate easily to any tag related feature set like language words that can be embedded into a space and modeled. Since the model doesn't use any pre-trained embeddings like Word2Vec and starts with randomly initialized embeddings, it intuitively translates to other use cases that don't emerge from natural language applications. The embeddings that would be learned by such a model could be invaluable in exploratory data analysis of entities and their latent space.

Bibliography

Bibliography

[1] Z. Ashktorab, C. D. Brown, M. Nandi, and A. Culotta, "Tweedr: Mining twitter to inform disaster response," in *ISCRAM*, 2014.

[2] K. Lim and A. Datta, *A topological approach for detecting Twitter communities with common interests*. Springer Berlin Heidelberg, 2013, pp. 23–43.

[3] S.-S. Ho, M. Lieberman, P. Wang, and H. Samet, "Mining future spatiotemporal events and their sentiment from online news articles for location-aware recommendation system," in *Proceedings of the First ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*, ser. MobiGIS '12. New York, NY, USA: ACM, 2012, pp. 25–32. [Online]. Available: http://doi.acm.org/10.1145/2442810.2442816

[4] K. W.-T. Leung, D. L. Lee, and W.-C. Lee, "Personalized web search with location preferences," in *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*. IEEE, 2010. [Online]. Available: https://doi.org/10.1109/icde.2010.5447911

[5] A. Rahimi, T. Cohn, and T. Baldwin, "A neural model for user geolocation and lexical dialectology," *CoRR*, vol. abs/1704.04008, 2017. [Online]. Available: http://arxiv.org/abs/1704.04008

[6] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, ser. EMNLP '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 1277–1287. [Online]. Available: http://dl.acm.org/citation.cfm?id=1870658.1870782

[7] Y. Ning, S. Muthiah, H. Rangwala, and N. Ramakrishnan, "Modeling precursors for event forecasting via nested multi-instance learning," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 1095–1104. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939802

[8] D. Kotzias, M. Denil, P. Blunsom, and N. de Freitas, "Deep multi-instance transfer learning," *CoRR*, vol. abs/1411.3128, 2014. [Online]. Available: http://arxiv.org/abs/1411.3128

[9] D. Kotzias, M. Denil, N. de Freitas, and P. Smyth, "From group to individual labels using deep features," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15. New York, NY, USA: ACM, 2015, pp. 597–606. [Online]. Available: http://doi.acm.org/10.1145/2783258.2783380

[10] A. Rahimi, T. Baldwin, and T. Cohn, "Continuous representation of location for geolocation and lexical dialectology using mixture density networks," *CoRR*, vol. abs/1708.04358, 2017. [Online]. Available: http://arxiv.org/abs/1708.04358

[11] S. Andrews, T. Hofmann, and I. Tsochantaridis, "Multiple instance learning with generalized support vector machines," in *Eighteenth National Conference on Artificial Intelligence.* Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 943–944. [Online]. Available: http://dl.acm.org/citation.cfm?id=777092.777234

[12] O. Maron and T. Lozano-Pérez, "A framework for multiple-instance learning," in *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, ser. NIPS '97. Cambridge, MA, USA: MIT Press, 1998, pp. 570–576. [Online]. Available: http://dl.acm.org/citation.cfm?id=302528.302753

[13] D. Kifer, "Attacks on privacy and definetti's theorem," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '09. New York, NY, USA: ACM, 2009, pp. 127–138. [Online]. Available: http://doi.acm.org/10.1145/1559845.1559861

[14] H. Kuck and N. de Freitas, "Learning about individuals from group statistics," *arXiv preprint arXiv:1207.1393*, 2012.

[15] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 1-2, pp. 31–71, 1997. [Online]. Available: http://lis.csail.mit.edu/pubs/tlp/multiple-instance-aij.pdf

[16] X. Xu and E. Frank, "Logistic regression and boosting for labeled bags of instances," in *Proc. of the PacificAsia Conf. on Knowledge Discovery and Data Mining.* Springer-Verlag, 2004, pp. 272–281.

[17] Z.-H. Zhou and J.-M. Xu, "On the relation between multi-instance learning and semi-supervised learning," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 1167–1174. [Online]. Available: http://doi.acm.org/10.1145/1273496.1273643

[18] N. Weidmann, E. Frank, and B. Pfahringer, "A two-level learning method for generalized multi-instance problems," in *In Proceedings of the Fourteenth European Conference on Machine Learning*, 2003.

[19] G. Liu, J. Wu, and Z.-H. Zhou, "Key instance detection in multi-instance learning," in *Proceedings of the Asian Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. C. H. Hoi and W. Buntine, Eds., vol. 25. Singapore Management University, Singapore: PMLR, 04–06 Nov 2012, pp. 253–268. [Online]. Available: http://proceedings.mlr.press/v25/liu12b.html

[20] S. Ray and M. Craven, "Supervised versus multiple instance learning: An empirical comparison," in *Proceedings of the 22Nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: ACM, 2005, pp. 697–704. [Online]. Available: http://doi.acm.org/10.1145/1102351.1102439

[21] T. Grtner, P. A. Flach, A. Kowalczyk, and A. J. Smola, "Multi-instance kernels," in *In Proc. 19th International Conf. on Machine Learning.* Morgan Kaufmann, 2002, pp. 179–186.

[22] Y.-F. L. J.-H. Hu and Y. J. Z.-H. Zhou, "Towards discovering what patterns trigger what labels," 2012.

[23] Y.-F. Li, J. T. Kwok, I. W. Tsang, and Z.-H. Zhou, *A Convex Method for Locating Regions of Interest with Multi-instance Learning.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 15–30.

[24] R. C. Bunescu and R. J. Mooney, "Multiple instance learning for sparse positive bags," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 105–112. [Online]. Available: http://doi.acm.org/10.1145/1273496.1273510

[25] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation," 01 2005.

[26] X. Zheng, J. Han, and A. Sun, "A survey of location prediction on twitter," *CoRR*, vol. abs/1705.03172, 2017. [Online]. Available: http://arxiv.org/abs/1705.03172

[27] F. Melo and B. Martins, "Automated geocoding of textual documents: A survey of current approaches," *Transactions in GIS*, vol. 21, no. 1, pp. 3–38, jun 2016. [Online]. Available: https://doi.org/10.1111/tgis.12212

[28] A. Woodruff and C. Plaunt, "Gipsy: Geo-referenced information processing system," vol. 45, 05 1994.

[29] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, "Web-a-where: Geotagging web content," in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '04. New York, NY, USA: ACM, 2004, pp. 273–280. [Online]. Available: http://doi.acm.org/10.1145/1008992.1009040

[30] Z. Cheng, J. Caverlee, and K. Lee, "You are where you tweet: a content-based approach to geo-locating twitter users," in *CIKM*, 2010.

[31] J. Liu and D. Inkpen, "Estimating user location in social media with stacked denoising auto-encoders," in *VS@HLT-NAACL*, 2015.

[32] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," vol. 521, pp. 436–44, 05 2015.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[34] V. Kůrková, "Kolmogorov's theorem and multilayer neural networks," *Neural Networks*, vol. 5, no. 3, pp. 501–506, Mar. 1992. [Online]. Available: http://dx.doi.org/10.1016/0893-6080(92)90012-8

[35] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, 1990, pp. 396–404.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[37] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," *arXiv preprint arXiv:1612.08083*, 2016.

[38] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[39] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013. [Online]. Available: http://arxiv.org/abs/1310.4546

[40] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 129–136.

[41] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[42] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[43] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[44] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[45] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[46] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[47] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[49] S. Roller, M. Speriosu, S. Rallapalli, B. Wing, and J. Baldridge, "Supervised text-based geolocation using language models on an adaptive grid," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, ser. EMNLP-CoNLL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 1500–1510. [Online]. Available: http://dl.acm.org/citation.cfm?id=2390948.2391120

[50] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[51] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ser. ACL '05. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 363–370. [Online]. Available: https://doi.org/10.3115/1219840.1219885

[52] A. Pardes, "A brief history of the ever-expanding tweet," Sep 2017. [Online]. Available: https://www.wired.com/story/a-brief-history-of-the-ever-expanding-tweet/

[53] F. Godin, B. Vandersmissen, W. De Neve, and R. Van de Walle, "Multimedia lab@ acl w-nut ner shared task: named entity recognition for twitter microposts using distributed word representations," *ACL-IJCNLP*, vol. 2015, pp. 146–153, 2015.

Curriculum Vitae

Sneha Nagpaul