

# Fine-grained Geolocation Prediction of Tweets with Human Machine Collaboration

Florina Dutt  
 florina.design@gatech.edu  
 Georgia Institute of Technology USA  
 Atlanta, GA

Subhajit Das  
 das@gatech.edu  
 Georgia Institute of Technology USA  
 Atlanta, GA

## ABSTRACT

Twitter is a useful resource to analyze peoples' opinions on various topics. Often these topics are correlated or associated with locations from where these Tweet posts are made. For example, restaurant owners may need to know where their target customers eat with respect to the sentiment of the posts made related to food, policy planners may need to analyze citizens' opinion on relevant issues such as crime, safety, congestion, etc. with respect to specific parts of the city, or county or state. As promising as this is, less than 1% of the crawled Tweet posts come with geolocation tags. That makes accurate prediction of Tweet posts for the non geo-tagged tweets very critical to analyze data in various domains. In this research, we utilized millions of Twitter posts and end-users domain expertise to build a set of deep neural network models using natural language processing (NLP) techniques, that predicts the geolocation of non geo-tagged Tweet posts at various level of granularities such as neighborhood, zipcode, and longitude with latitudes. With multiple neural architecture experiments, and a collaborative human-machine workflow design, our ongoing work on geolocation detection shows promising results that empower end users to correlate relationship between variables of choice with the location information.

## KEYWORDS

Classification, Regression, Sequence data, Twitter API, Transformers, Natural language processing, Human-centered Machine Learning

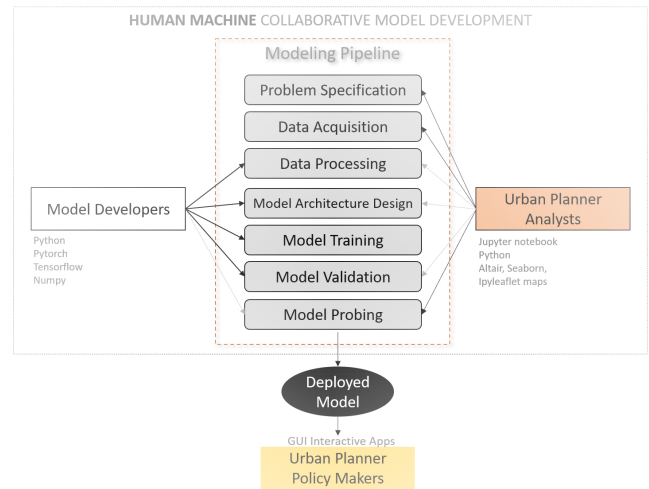
### ACM Reference Format:

Florina Dutt and Subhajit Das. 2021. Fine-grained Geolocation Prediction of Tweets with Human Machine Collaboration .

## 1 INTRODUCTION

Machine learning (ML) has been effectively used in many real-world data analytic problem scenarios (e.g., in marketing, finance, healthcare, etc. [27]). In many domains, analysts' intend to make sense of peoples' opinion to make informed business decisions. For example, in location based advertisements user reviews are analysed with location, in citizen opinion mining tasks [9, 19] Twitter posts are used to analyse topics and entities with geolocation [32]. In the literature and in real world analysis problems, Twitter posts (often crawled using Python's Tweepy API [5]) are a good data source for conducting such analysis. However, geolocated Tweets are very rare (less than 1%) [25] which makes it impossible to utilise the

entire data corpus for analysis. In this paper, we present our ongoing work to predict Twitter posts' geolocation at a granular level such as neighborhood, zipcode, and longitude with latitude values. While geolocating Twitter posts has been researched in the past [29, 31, 35], granular location prediction [10, 20] is rare and under-explored, which is often critical for many domains. In this work, we collaborate with urban planning analysts to not only assess the success of our modeling task by an iterative modeling pipeline but also incorporate their valuable domain expertise in making modeling decisions to refine model performance (see Figure 1). End-users (urban planners) explore and analyse the model output results using interactive visualizations in a Jupyter notebook environment. In the process they spot check results (of location prediction) and provide domain-specific critical feedback, that is programmatically implemented to: (1) clean and transform data, (2) re-design the models' architecture, and (3) extract guidelines to inform processes to further collect training data.



**Figure 1: Workflow adopted in this collaborative model development between model developers and domain experts.**

We present a set of best performing models in addition to an in-depth summary of various model experimentation results. We discuss the successes and failures of various modeling approaches for our problem, some of which were partly inspired from the literature (see section 2), while many other algorithmic approaches were driven by feedback from domain experts (urban planners) as they probed the trained model in a Jupyter notebook environment with interactive visualizations. In summary, we contribute: (1) A novel text modeling approach informed with feedback from domain expert users to predict geolocation information for non geo-tagged

Tweet posts at zipcode, neighborhood, and longitude with latitude granularity. (2) Results including successes and failures from an on-going work with urban planner analysts of a real world ML problem where geolocated Tweets are needed to assess citizens' opinion on various topics critical to craft new urban policies.

## 2 RELATED WORK

Predicting geolocations from Tweet posts has been investigated extensively. Luke et al. built an interactive system to predict geolocation for Tweets at the level of a city [29]. Yasuhide et al. showed integrating text, meta data and user network to build a neural network with attention mechanism to predict location information [24]. These surveys further provide a comprehensive overview of location detection research from Twitter posts [31, 35]. Miyazaki et al. employed graph embedding on a knowledge graph to build an end to end geolocation classifier [25]. Tien et al. discussed MENET, a multi entry neural network architecture combining deep neural network and multi view learning that outperformed three publicly available datasets on city level prediction (metric used was Acc@161) [12]. Chieh et al. explored multi-headed self attention model for text representation to predict location at the level of a city [13]. While these works were effective, they were not granular location prediction methods to predict a Tweets location. Aligned to this, Wen-haw et al. used a hidden markov model to geolocate tweets to a closest venue such as restaurants, shops, and others [10]. Matsuno et al. proposed a technique to predict a Twitter users' predicted location visit to a block level [22]. Inspired from these works, we further geolocate tweets to a zipcode, neighborhood or longitude and latitude.

Li et al. discussed the application of fine-grained geolocation based on extraction of location related entities from non-geotagged Tweet posts [20]. Iso et al. deployed a convolutional mixture density network to predict Tweets' geolocation along with ambiguity in prediction based on a probability distribution. Their best model predicted a Tweet with an mean error of 91 miles [14]. Zola et al. estimated a users' home location using gaussian mixture models and the DBSCAN clustering approach [36]. Li et al. showed that their model outperformed others when trained using diverse data sources including textual content, user location, place labels and bounding box of place [18]. Mircea et al. prototyped a classification, geolocation and interactive visualization of COVID-19 tweets. Their dashboard helped users to geolocate Tweets by fine tuning a novel L2 classification layer [23]. These works inspired us to: (1) continue research in this direction, and (2) fill the missing gap of more direct human machine collaboration in model development to predict fine-grained geolocation information from social media data.

## 3 TECHNIQUE

### 3.1 Problem specification

**Dataset:** The data was crawled from Twitter using Python's Tweepy library [5] between *Jan 2018* to *Dec 2020*, within 100 mile radius of Atlanta, Georgia, USA. In total, there were 10 million Tweets, out of which only 90000 Tweets were geo-tagged. We filtered the data using keywords specific to urban planning domain (provided by the

urban planners), to sample an useful set of 2 million Tweet posts. This also helped discard a large set of redundant Tweet posts related to various kinds of product/service advertisement including jobs, restaurants, product promotions, and others. We set 80000 data samples for training set, 10000 for validation set, and the rest of the data (2 million - 90000 samples) for the hold out/test set. In addition, we also validate our models' performance on 200000 Tweets from the US Global dataset [1] comparing geolocation prediction over Tweets from multiple cities in USA (see section 3.3 and Figure 2). Each Tweet from both datasets contained attributes such as *Tweet-text*, *Created-at*, *Num-Hashtags*, *Username*, *Retweet-count*, and others. For geo-tagged tweets, the data also contained the attributes *Longitude* and *Latitude*.

**Pre-processing:** Prior to inputting the data to the modeling pipeline, the data is pre-processed and transformed in ways which improves the performance of the model. Each Tweets text can be expressed as a list of words  $W$ . We pre-process  $W$  using standard natural language processing methods including stemming, lemmatization, removal of stop words and special characters, and limiting text length (user-specified hyperparameter) [11, 26]. Further pre-processing is applied as users interact and explore the data by filtering relevant Tweets, specifying more relevant class labels, adjusting hyperparameters, probing models through visualizations as seen in Figure 1.

**Task:** With the crawled data, urban planners intended to analyse peoples' sentiment, as expressed in their opinions through Twitter posts. In this human machine collaboration, they strive to design a deep learning model that can predict geolocation of non geo-tagged Tweets with highest accuracy. More importantly, their data analytic workflow required the geolocation prediction to be fine-grained, at-least 30 miles or less in prediction error.

### 3.2 Model design and selection

In the following we describe a series of model architecture and hyperparameter settings (see Table 2), each designed for the specified task (classification or regression) by the user (urban planners). In general, our approach was to design an end-to-end ML modeling pipeline that given an input text sequence (e.g., a Tweet post) predicts the geolocation of the post such as zipcode, neighborhood or longitude with latitude as specified by the user.

**Longitude and Latitude prediction:** Initially, we posed the problem as a regression task, as we were motivated to predict longitude and latitude. We utilised a 1-dimensional convolution based CNN for sentence classification model ( $S$ ) as described by Kim et al. [16]. We trained  $S$  with a wide range of kernels of size 2, 4, 8, 16, 32, 64 and with 128 channels each. We applied max pooling after each convolutional layer. In addition, we added 2 fully connected layer. Other hyperparameters included learning rate (varied between 0.001 to 0.09 and batch size, varied between 128 to 2048). Finally, with 70 epochs, 0.005 learning rate, and batch size of 256 we observed better performing results. In this model architecture, we modified the output layer to contain two nodes predicting longitude and latitude of a given input Tweet post. We utilized the pairwise distance loss function [28] to assess the performance of this model ( $\sqrt{\sum_0^n (p_i - q_i)^2}$ ). While this model performed relatively well on

the training set (acc@30miles: 47.28%, average distance between predicted location < 38.23 miles, loss: 0.45), the performance was very poor on the validation set (acc@30miles: 12.11%, average distance between predicted location <= 250 miles, loss: 9.453). We asked domain experts to explore word/token distribution present in Tweets with respect to longitude and latitude through visualizations in the Jupyter notebook, to help understand if there was something peculiar about the input data and the target label distribution. Upon probing, they found that there was a lack of any correlation between the urban planning relevant tokens (including bi-grams and tri-grams) (e.g., "green park", "biking along the river", "stuck in traffic") with the longitude and latitude information.

To overcome the inherent noise in the data, we were inclined to model character-level nuances to predict fine-grained location. We utilized the model architecture of a character-level CNN from the work of McKenzie et al. [7] and others [21]. However, we did not want to lose the information learned through the word level CNN model as described above. In this, we were motivated by the approach shown by Ross et al. [13] in predicting cities. In their model architecture they concatenated the input from two parallelly trained models to make predictions. We first trained the character CNN model  $C$  and retrieved the final fully connected layers' output as  $e$ . Next, we trained the sentence level modified CNN model  $S$  (as described above); however, before the final layer making the prediction, we concatenated the output from the fully connected layer with  $e$ . This model architecture design helped to feed character level nuances into the modeling problem. This approach (CCH) improved the validation sets' performance (acc@30miles: 28.15%, average distance between predicted location <= 100 miles, loss: 2.322). Furthermore, we replaced the model  $S$  with a word level LSTM model [34] modified with an attention layer  $A$ , inspired by the work of Vaswani et al. [30] in the above pipeline. We utilized attention mechanism to calculate a soft alignment score between the hidden states for each token and the final hidden state representation in the LSTM model.  $A$  first computes weights for each sentence in LSTM output and finally computes the updated hidden state. With this model (CCH-A,  $T$ , containing  $C$  and  $A$ ) we noticed a slight improvement in performance over the validation set (acc@30miles: 36.09%, average distance between predicted location <= 85.43 miles, loss: 0.322).

**Zipcode prediction:** Models trained to predict *Longitude* and *Latitude* showed less than satisfactory results. Based on feedback from urban planners, we decided to predict zipcodes. To retrieve zipcodes for the geo-tagged data, we utilized Arcgis' reverse geocoding API for Python [2], which given an input tuple of longitude, latitude pairs, returned a JSON object containing the address of the location. This JSON object included street name, county, neighborhood name, and the zipcode. The data contained 727 zipcodes from the state of Georgia, USA, as target labels in the input data. Now our task was to build a classifier instead of a regressor.

Our next model architecture is based on a modified transformer model (with multi-headed attention layer) [30] to predict the zipcode of a Tweet (categorical target variable). This model (MH)  $M$  is different from a conventional transformer model in the following ways: (1) we discarded the decoder from the transformer model, (2) we removed residual connections, layer normalization, and the

layer masking (as its' a classification task, not a language modeling problem), and (3) we employed a multi-headed attention with position-wise feedforward encodings. These changes are inspired by the implementation referred here [4]. Since this was a classification task to predict zipcodes the final layer incorporated a cross entropy loss with soft-max activation function ( $-\sum_0^n y_i \log(J(f_0(x_i)))$ ),  $J$  is softmax function). When trained, this model  $M$  performed very highly on the training set (acc: 83.32%, loss: 0.12) but poorly on the validation set (acc: 22.23%, loss: 1.65). We specified batch-size: 512, learning-rate: 0.004, number of encoders: 3, embedding dimension: 1024, division-factor: 128, number-epochs: 50 as hyperparameters to this model, and optimized using the Adam optimizer [8, 17]. Aiming to reduce overfitting, we specified a drop out rate of 0.3, reduced batch-size to 64 and utilised a learning rate decay function. However, this did not improve the generalizability of the model as per expectation.

We brainstormed with the urban planners as we reviewed the text content of the Tweets and the set of users who were the authors of the Tweets in our data corpus. Upon investigation, we found that there were many users who were posting content more frequently than others. We also found that they shared similar content especially within a close time range (say within a weeks time) from a nearby geolocation (e.g., within 10 miles of each post). Based on this insight, we constructed a username dictionary object  $U$  (from the training set) that stored per user  $u_i$ : (1) posted Tweet Ids  $w_1, w_2, w_3, \dots$ , Tweet creation date/time  $tm_1, tm_2, tm_3, \dots$ , and their geolocation information  $l_1, l_2, l_3$ . Next, only for validation set, during prediction we query for the username from  $U$ . If the username is in the dictionary we retrieve all the tweets with the tweet ids  $w_1, w_2, w_3, \dots$ , for that user. Next, based on the creation date of the Tweet in the validation set, we sample a subset of tweets  $f$  from  $U$  (specified by a given time range by users which is a hyperparameter, e.g., find Tweets within 2 days of post). Finally, we sample the first  $k$  (an hyperparameter) tokens of each tweet in  $f$  and concatenate with the queried Tweets text content. The intuition of this approach directly stemmed from the observations made by the urban planners upon analysing the training corpus. Their insight was that users usually post on similar topics/content within a close time range from close-by geolocations. For example, posts about review of a restaurant within a weeks time tend to contain similar text content. Furthermore Wen-haw et al. [10] followed a similar approach in predicting fine-grained geolocation for non geo-tagged Tweets. However, unlike their approach, we do not sample users' Tweet post based on their visit to a same venue such as restaurant, shops, museum etc. Using this approach (MH-U), we observed an increase in validation set performance of zipcode prediction (acc: 47.30%, loss: 0.72).

Next, motivated to improve the validation sets' accuracy, we further investigated other approaches that can be adopted using text content from usernames in the corpus. In this approach, instead of building a username dictionary object, we first prepared an embedding matrix of all the Tweets in the training corpus  $e$ . To generate the embeddings per Tweet, we utilized Tensorflows' universal sentence encoder [6]. Next we clustered the embedding matrix using the KMeans clustering model [15] with a user-specified number of clusters, which is a hyperparameter. As we train the model, it first queries the cluster membership of the username of a given

Tweet. Next from  $e$  it samples  $nc$  (a hyperparameter specified by user) closest matching tweets (defined by cosine similarity metric) from the found cluster as  $g$ . The model consumes  $g$  by passing it through its' encoder layer, and then concatenating the results of the encoder layer with the encoded representation of the input text as  $o$ . Finally  $o$  is passed through a fully connected layer to make zipcode prediction. We were happy to see a significant increase in the models validation set performance (acc: 59.43%, loss: 0.18) with this approach (MH-C). Upon inspection by the urban planners, we observed that the zipcode distribution was skewed in the validation set. With stratified sampling on the validation set (MH-C-S) and tuned hyperparameters (see Table 2) the performance further increased to acc: 67.30%, loss: 0.09.

**Neighborhood prediction:** While we were successful to train a model with significantly better (compared to longitude level) prediction results with respect to zipcodes, urban planners required the model to also predict neighborhood of a given tweet for their analysis process. Their rationale was that neighborhoods are much more comprehensive and relatable as compared to zipcodes to make sense of analysis results in their domain. In addition, typically the sizes of the zipcodes show a high variance, some zipcodes can be very small in area, while many others cover a large geographic region. This variance in the zipcode size further creates ambiguity in the prediction. Neighborhood level sentiment analysis of Tweets allows urban planners to compare these sentiment scores from peoples' opinion with other factors such as built environment characteristics such as street lights, side walk depth, and others. As mentioned before, the JSON object retrieved as address from the ArcGIS Api contained neighborhood data. However, most of the data samples in the JSON object had "null" values for the neighborhood key. To solve this problem, with the help of the urban planners we were able to use a neighborhood shape file [3] for the city of Atlanta metropolitan area in Georgia, USA. For each geo-tagged Tweet, using ArcGIS' python library Arcpy, we spatially joined the neighborhood shape file to fetch the neighborhood label per Tweet. In total there were 102 neighborhoods in our data corpus.

To predict the neighborhood, we utilized a similar pipeline as we did to predict zipcodes, that included: (1) training a multi-headed attention model, and (2) creation of a user dictionary object to add text content to Tweets in the validation set. However, in the case of the neighborhood label prediction task, we observed a class imbalance. We thus over-sampled the minority class to balance the data. However, this significantly increased the training time and the memory requirement in our pipeline. We thus introduced a *sample-factor* hyperparameter, that ranged between 0 (means no oversampling) to 1 (means complete oversampling with each class balanced). We observed that the value 0.65 worked well for our use case and task. Next, we utilized the glove embeddings instead of the universal sentence encoders to create the embedding matrix  $e$ . Glove embedding was not only faster to compute but also resulted in improved performance. With this model architecture design we were able to significantly improve validation sets' performance (acc: 71.32%, loss: 0.003).

**Table 1: Various model performance results on our Twitter data.**

Model	Acc. (valid)	Loss	Key-Params	Target
1D-CNN	12.11%	9.453	kernels: (2 - 64) fc layers: 2 l-rate: 0.04	Long, Lat
CCH	28.15%	2.322	kernels: (4 - 32) fc layers: 1 l-rate: 0.09	Long, Lat
CCH-A	36.09%	0.322	kernels: (2 - 64) fc layers: 3 l-rate: 0.005	Long, Lat
MH	22.23 %	1.65	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.3	Zipcode
MH-U	47.30%	0.72	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.3 k-token: 16 time: 72 hours	Zipcode
MH-C	59.43%	0.182	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.3 num-clus: 4 samples: 20	Zipcode
MH-C-S	67.30%	0.091	encoders: 4 emb-dim: 3 div-fac: 256 drop-out: 0.33 num-clus: 9 samples: 35	Zipcode
MH-N-E	71.32%	0.003	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.1 sam-fac: 0.65	Neighbor -hood
MH-N	58.34%	0.194	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.1	Neighbor -hood

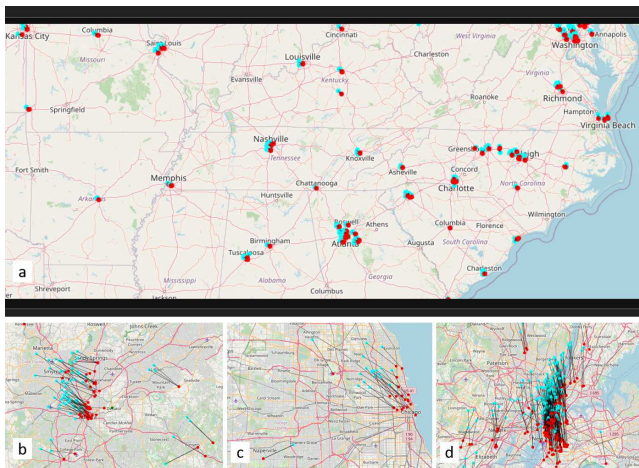
### 3.3 Generalizing models

From the best set of models that we trained, we also tested their performance on the US Global Tweets dataset [1] (see Figure 2). Of the 200000 Tweets, we set 150000, 20000, and 30000 for training, validation, and test/hold out set respectively. We used the CCH-A model to predict longitude and latitudes and observed satisfactory performance (acc@30miles: 32.12%, average distance between predicted location  $\leq$  109.66 miles, loss: 0.853). We tuned the hyperparameters to observe a little improvement in performance (acc@30miles: 39.59%, average distance between predicted location  $\leq$  81.01 miles,

**Table 2: Various model performance results on US Global Twitter data [1].**

Model	Acc. (valid)	Loss	Key-Params	Target
CCH-A	39.59%	0.632	kernels: (4 - 16) fc layers: 2 l-rate: 0.046	Long, Lat
MH-C-S	65.03%	0.09	encoders: 3 emb-dim: 3 div-fac: 256 drop-out: 0.33 num-clus: 9 samples: 35	Zipcode
MH-N-E	72.84%	0.01	encoders: 3 emb-dim: 3 div-fac: 128 drop-out: 0.1 sam-fac: 0.65	Neighbor-hood

loss: 0.632). Next we trained the MH-C-S model to predict zipcodes (in total 5000 zipcodes). Upon investigating further, we remove zipcodes with less than 10 Tweet samples. The final dataset had 170000 samples (130k training, 20k validation, and 20k test set) with 900 zipcodes. When trained we observed acc: 65.03%, loss: 0.09 on the validation set. Finally we trained the model MH-N-E on this data and observed acc: 68.66%, loss: 0.11 on the validation set. Upon changing the oversampling hyperparameter and discarding low frequency neighborhood Tweet samples we observed a bit higher performance (acc: 72.84%, loss: 0.01) on the validation set. With this excersize, we ensured that the model designs chosen for this analytic pipeline were generalizable to other datasets as well.



**Figure 2: Visualization in Jupyter notebook of predicted geolocations (shown in blue dot) compared to ground truth (shown in red dot) for: a. USA, b. Atlanta, c. Chicago, d. New York**

## 4 EXPERIMENTS

In this section, we describe a set of ablation studies to learn the impact and role of various components in our modeling pipeline. The data used in these studies are same as the one on which we trained models (i.e, Twitter posts within 100 mile radius of Atlanta between 2018 to 2020).

**User data dictionary:** We were inquisitive to understand the relevance of the user data dictionary in modeling the location prediction problem. To that end, we ran an ablation study in which we varied the parameter that controlled the number of Tweets to sample from the user object. It ranged from 0 (meaning no Tweet will be sampled) to 1 (meaning all Tweets from that user will be sampled). When the parameter was set to 0 we found very poor validation set accuracy (acc: 22.2%), while with the parameter set to 0.5 the validation set accuracy maximized (acc: 45.22%). Upon further setting the parameter to 1, we found mild reduction in performance (acc: 41%). Thus we understand that there is a sweet spot in which relevant text from the same user when added to the Tweet leads to increasing the performance. Adding every possible Tweets' content (of the same user) leads to adding noise, which causes drop in the models' performance. We performed a similar study with the parameter that controlled the time range within which relevant Tweet from the same user will be sampled. We ranged it between 0 (means no Tweet will be sampled) to 1 (every Tweet of the user will be sampled irrespective of the time). We found the value of the parameter when set to 0.44 (meaning within a span of 3-4 days) led to the maximum validation set accuracy. When the parameter was set at 1, it caused adding noise, and lost the mapping of the input text to the location label (in this case the zipcode).

**Clustering effect:** Furthermore, we ran an ablation study to understand the effect of the number of clusters on validation sets accuracy. We trained the model predicting zipcodes with number of clusters ranging between 1 to 15, incremented with step size 2. We found that the accuracy was quite low with *num-cluster* = 1 (acc: 32.24%). It increased to the maximum value with *num-cluster* = 6 (acc: 65.34%). Upon further increasing the *num-clusters* the accuracy dropped to 61.23%. We learned that when there is only one cluster there is no sense of similarity between the Tweets and thus each training input is injected with noise rather than text from similar other data samples. Upon increasing the number of clusters to an optimal value, the training input is concatenated with the encoded representation of similar tokens that map with the location label. However, when the number of clusters is way too many, then there are not adequate number of similar Tweets to look into to inject into the modeling problem. It is noteworthy, to mention that that the clustering based embedding is added only during the training phase of the model. During validation we only use the input Tweet text.

**Neighborhood embedding:** Finally we were curious to learn about the relevance of the pre-computed neighbor embedding of Tweets based on cosine similarity. When we included the embeddings in the model training process, we achieved a validation accuracy of 71.32% on the neighborhood prediction task. When we trained the model without the embeddings, we found that the validation set accuracy dropped to 58.34%. This confirmed that the embeddings'

helped the model learn the data better to predict the neighborhood codes. The parameter that controlled the number of similar Tweets to use, was set based on trial and error till we achieved the best performance results for our dataset. In the future, we need further studies to understand the impact of this value to the modeling problem.

## 5 DISCUSSION AND LIMITATION

**User interaction and model exploration:** The human machine collaborative modeling pipeline presented in the paper, needed spot checking model results and conducting exploratory data analysis to make sense of the results. User interaction was critical in this process, which was deployed using interactive visualizations in Jupyter notebook, leveraging various Python libraries including altair, seaborn, ipyleaflet, matplotlib, and others (see Figure 1). This was adequate for domain expert users who had strong grounding on data analysis and beginner level understanding of ML modeling. However, for domain experts who are not skilled/trained as analysts, we understand that we need explicit visual interfaces that abstracts user interaction to empower them in the modeling process. With the success of this human machine collaboration in building better performing models for urban planners, we plan to further develop interactive visual interfaces to support this process as the next step. Yang et al. confirmed that it is crucial to attain the right level of abstraction in scalable user interaction to employ robust and successful human machine collaborative model development [33]. In this work, Jupyter notebooks were sufficient to provide the scalable user interaction needed by the urban planner analysts'. However, moving forward we need more further research in abstracting user interactions and scalable interfaces that can seamlessly empower co-developing models as users interact with the model results.

**User name dictionary:** Most of the better performing models presented in this paper, relies on the creation of a user name dictionary. Our hypothesis is that, when queried with a new Tweet with username as the meta data, our technique would augment the Tweet text with other text content retrieved from the user name dictionary object. This approach worked very well for us as the exploratory data analysis by domain experts revealed that there are many frequently posting users. However, we are aware that this approach may be less effective on a dataset on which the user name dictionary is sparse. In that case, we may rely on other meta data content that we have not investigated in this work including hash tags, Tweet mentions, ReTweet counts, and others. In general, we have observed that frequently occurring user names are fairly common and there's a high likelihood of creating a dense user name dictionary that helps improve the models' prediction.

**Fine-grained prediction:** In this work, urban planners were motivated to build models for fine-grained Tweet prediction. In that endeavor, we started with latitude and longitude prediction. While we observed some success in their prediction, given the noise and ambiguity in short text posts in Twitter, improving accuracy closer to the Tweet posts less than 30 miles (as desired by end-users) seemed a bit too ambitious. Courtesy to the collaboration with the domain experts, we were able to adjust our analytical goal

from longitudes to zipcode, and then to neighborhood level prediction. Noteworthy to mention that both zipcodes and neighborhood level predictions are still fine-grained compared to what we observed from the literature that most of the previous work in geolocation prediction have been at the level of country or city. In the future, we plan to continue our current research in predicting fine-grained Tweet geolocation based on the lessons learned from the conducted experiments.

## 6 CONCLUSION

In this paper, we demonstrate our investigation into modeling fine-grained geolocation prediction of non geo-tagged Tweets at multiple levels of granularity including zipcodes, neighborhoods, and longitude with latitude information. In the modeling process, we include humans (end-users) to directly adjust model architecture providing an opportunity to feed their domain expertise to better solve their analytical problem. We collaborate with urban planners, in which we support them geolocate Tweets to better analyse peoples' sentiment as expressed through Twitter posts. Through a set of experiments on a variety of deep neural network architectures, we show our data analysis pipeline supports training a robust model that generalizes well to unseen Tweet posts.

## REFERENCES

- [1] 2021. 200,000 USA geolocated tweets. Free Twitter Dataset. <http://followthehashtag.com/datasets/free-twitter-dataset-usa-200000-free-usa-tweets/>. Accessed: 2021-05-18.
- [2] 2021. ArcGIS API for Python. <https://developers.arcgis.com/python/guide/reverse-geocoding/>. Accessed: 2021-03-18.
- [3] 2021. City of Atlanta Neighborhood Statistical Areas. [https://opendata.atlantaregional.com/datasets/d6298dee8938464294d3f49d473bcf15\\_196/explore?l=1](https://opendata.atlantaregional.com/datasets/d6298dee8938464294d3f49d473bcf15_196/explore?l=1). Accessed: 2021-05-18.
- [4] 2021. Text classification Pytorch. <https://github.com/prakashpandey9/Text-Classification-Pytorch>. Accessed: 2021-02-25.
- [5] 2021. Tweepy API. <https://docs.tweepy.org/en/latest/api.html>. Accessed: 2021-02-25.
- [6] 2021. Universal Sentence Encoder. [https://www.tensorflow.org/hub/tutorials/semantic\\_similarity\\_v1](https://www.tensorflow.org/hub/tutorials/semantic_similarity_v1). Accessed: 2021-05-18.
- [7] Benjamin Adams and Grant McKenzie. 2018. Crowdsourcing the character of a place: Character-level convolutional networks for multilingual geographic text classification. *Transactions in GIS* 22, 2 (2018), 394–408. <https://doi.org/10.1111/tgis.12317> arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12317>
- [8] Somenath Bera and Vimal K. Shrivastava. 2020. Analysis of various optimizers on deep convolutional neural network model in the application of hyperspectral remote sensing image classification. *International Journal of Remote Sensing* 41, 7 (2020), 2664–2683. <https://doi.org/10.1080/01431161.2019.1694725> arXiv:<https://doi.org/10.1080/01431161.2019.1694725>
- [9] Shelley Boulianne. 2015. Social media use and participation: a meta-analysis of current research. *Information, Communication & Society* 18, 5 (2015), 524–538. <https://doi.org/10.1080/1369118X.2015.1008542> arXiv:<https://doi.org/10.1080/1369118X.2015.1008542>
- [10] Wen-Haw Chong and Ee-Peng Lim. 2019. Fine-Grained Geolocation of Tweets in Temporal Proximity. *ACM Trans. Inf. Syst.* 37, 2, Article 17 (Jan. 2019), 33 pages. <https://doi.org/10.1145/3291059>
- [11] Subhajt Das and Florina Dutt. 2020. InMacs: Interactive modeling and comparison of sentiments from sequence data. In *1st Workshop on Data Science with Human in the Loop, DaSH@KDD 2020, San Diego, California, USA, August 24, 2020*, Eduard C. Dragut, Yunyao Li, Lucian Popa, and Slobodan Vucetic (Eds.). <https://drive.google.com/file/d/1kbhBs5MtaiZHOeiVUUu0CHu0OGHjJrZ/view>
- [12] T. Do, Duc Minh Nguyen, Evaggelia Tsiligianni, Bruno Cornelis, and N. Deligiannis. 2017. Multiview Deep Learning for Predicting Twitter Users' Location. *ArXiv abs/1712.08091* (2017).
- [13] Chieh-Yang Huang, Hanghang Tong, Jingrui He, and Ross Maciejewski. 2019. Location Prediction for Tweets. *Frontiers in Big Data* 2 (2019), 5. <https://doi.org/10.3389/fdata.2019.00005>
- [14] Hayate Iso, Shoko Wakamiya, and Eiji Aramaki. 2017. Density Estimation for Geolocation via Convolutional Mixture Density Network. *CoRR abs/1705.02750*

- (2017). arXiv:1705.02750 <http://arxiv.org/abs/1705.02750>
- [15] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. 2002. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 7 (2002), 881–892. <https://doi.org/10.1109/TPAMI.2002.1017616>
- [16] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. <https://doi.org/10.3115/v1/D14-1181>
- [17] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. <http://arxiv.org/abs/1412.6980> cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.
- [18] Bingnan Li, Zi Chen, and Samsung Lim. 2021. Geolocation Inference Using Twitter Data: A Case Study of COVID-19 in the Contiguous United States. In *Geographical Information Systems Theory, Applications and Management*, Cédric Grueau, Robert Laurini, and Lemonnia Ragia (Eds.). Springer International Publishing, Cham, 119–139.
- [19] Dawei Li, Yujia Zhang, and Cheng Li. 2019. Mining Public Opinion on Transportation Systems Based on Social Media Data. *Sustainability* 11, 15 (2019). <https://doi.org/10.3390/su11154016>
- [20] Yongjun Li, Wenli Ji, Yao Deng, and Xing Gao. 2020. An Entity-Based Fine-Grained Geolocalization of User Generated Short Text. *IEEE Access* 8 (2020), 219114–219123. <https://doi.org/10.1109/ACCESS.2020.3042813>
- [21] Jingxue Liu, Fanrong Meng, Yong Zhou, and Bing Liu. 2017. Character-Level neural networks for short text classification. In *2017 International Smart Cities Conference (ISC2)*. 1–7. <https://doi.org/10.1109/ISC2.2017.8090812>
- [22] Shogo Matsuno, Sakae Mizuki, and Takeshi Sakaki. 2020. Improved Advertisement Targeting via Fine-Grained Location Prediction Using Twitter. In *Companion Proceedings of the Web Conference 2020 (Taipei, Taiwan) (WWW '20)*. Association for Computing Machinery, New York, NY, USA, 527–532. <https://doi.org/10.1145/3366424.3382118>
- [23] Andrei Mircea. 2020. Real-time Classification, Geolocation and Interactive Visualization of COVID-19 Information Shared on Social Media to Better Understand Global Developments. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*. Association for Computational Linguistics, Online. <https://doi.org/10.18653/v1/2020.nlpccovid19-2.37>
- [24] Yasuhide Miura, Motoki Taniguchi, Tomoki Taniguchi, and Tomoko Ohkuma. 2017. Unifying Text, Metadata, and User Network Representations with a Neural Network for Geolocation Prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 1260–1272. <https://doi.org/10.18653/v1/P17-1116>
- [25] Taro Miyazaki, Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2018. Twitter Geolocation using Knowledge-Based Methods. In *NUT@EMNLP*.
- [26] Richard A. Plunz, Yijia Zhou, Maria Isabel [Carrasco Vintimilla], Kathleen Mckeown, Tao Yu, Laura Ugucioni, and Maria Paola Sutto. 2019. Twitter sentiment in New York City parks as measure of well-being. *Landscape and Urban Planning* 189 (2019), 235 – 246. <https://doi.org/10.1016/j.landurbplan.2019.04.024>
- [27] Foster Provost and Ron Kohavi. 1998. On Applied Research in Machine Learning. In *Machine learning*. 127–132.
- [28] Yuxiang Qin, Chungang Yan, Guanjun Liu, Zhenchuan Li, and Changjun Jiang. 2020. Pairwise Gaussian Loss for Convolutional Neural Networks. *IEEE Transactions on Industrial Informatics* 16, 10 (2020), 6324–6333. <https://doi.org/10.1109/TII.2019.2963434>
- [29] Luke S. Snyder, Morteza Karimzadeh, Ray Chen, and David S. Ebert. 2019. City-Level Geolocation of Tweets for Real-Time Visual Analytics. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery (Chicago, IL, USA) (GeoAI 2019)*. Association for Computing Machinery, New York, NY, USA, 85–88. <https://doi.org/10.1145/3356471.3365243>
- [30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>
- [31] Shuai Xu, X. Fu, J. Cao, Binghang Liu, and Z. Wang. 2020. Survey on user location prediction based on geo-social networking data. *World Wide Web* 23 (2020), 1621–1664.
- [32] Zhiheng Xu, Long Ru, Liang Xiang, and Qing Yang. 2011. Discovering User Interest on Twitter with a Modified Author-Topic Model. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, Vol. 1. 422–429. <https://doi.org/10.1109/WI-IAT.2011.47>
- [33] Yiwei Yang, E. Kandogan, Yunyao Li, P. Sen, and Walter S. Lasecki. 2019. A Study on Interaction in Human-in-the-Loop Machine Learning for Text Analytics. In *IUI Workshops*.
- [34] Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-State LSTM for Text Representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 317–327. <https://doi.org/10.18653/v1/P18-1030>
- [35] Xin Zheng, Jialong Han, and Aixun Sun. 2018. A Survey of Location Prediction on Twitter. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1652–1671. <https://doi.org/10.1109/TKDE.2018.2807840>
- [36] Paola Zola, Paulo Cortez, and Maurizio Tesconi. 2020. Using Google Trends, Gaussian Mixture Models and DBSCAN for the Estimation of Twitter User Home Location. In *Computational Science and Its Applications – ICCSA 2020*, Osvaldo Gervasi, Beniamino Murgante, Sanjay Misra, Chiara Garau, Ivan Blečić, David Taniar, Bernady O. Apduhan, Ana Maria A. C. Rocha, Eufemia Tarantino, Carmelo Maria Torre, and Yeliz Karaca (Eds.). Springer International Publishing, Cham, 526–534.