CS230

Using pre-Q sequences of reddit posts to predict user-level QAanon participation

Lillian Ma & Stephanie Vezich
Department of Computer Science
Stanford University
{ylma, isvezich}@stanford.edu

Abstract

The same user in general purpose online communities such as reddit may post in a variety of channels ranging from quotidian interest forums to political extremist groups. A user that does eventually join extremist groups typically has some history of participation preceding that action. The goal of the current work is to investigate whether it is possible to predict extremist group membership from the language of this preceding participation. Using the reddit Pushshift API, we collect all posts authored by a sample of users who either later joined QAnon subreddits or did not. We then use language embeddings to predict class membership, achieving 79% accuracy and 0.80 F1-score. We find superior performance using pre-trained sentence embeddings relative to task-specific word embeddings. We also find superior performance using sequence models, particularly LSTM, relative to an MLP baseline. These findings suggest that the sequence of a user's posts may encode their development of extremist attitudes, and provide a method for online community platforms to identify and potentially intervene on at-risk users.

1 Introduction

The impact of conspiracy theories has been particularly acute in recent years, with deleterious effects ranging from lowered public belief in scientific findings [2] to mobilizing violent extremist groups [1] [3]. One especially influential conspiracy theory movement is QAnon, which posits that "a group of Satan-worshiping elites who run a child sex ring are trying to control our politics and media" and has spread misinformation about a range of topics including Covid-19, Black Lives Matter, and the 2020 U.S. presidential election [12]. A December 2020 poll estimated at least 17% of Americans believe QAnon claims [10]. Moreover, the U.S. Department of Homeland Security and the Federal Bureau of Investigation have flagged certain QAnon followers as domestic violent extremists, highlighting the serious safety threat this group poses [11].

Though discussion about Q drops–or posts from an anonymous user known as Q–has appeared on several messaging platforms, reddit was among the most mainstream [15]. Reddit is a particularly interesting setting to examine because it tends to attract a fairly broad audience who may not initially hold extremist beliefs. Yet, some subset of these users became increasingly engaged in the QAnon community from the first Q-drop on October 28, 2017, on 4chan, throughout the next several years. If it is possible to identify a priori who might develop extremist beliefs, it may also be possible to intervene and potentially prevent domestic terrorist acts. For example, one might imagine modifying recommendation algorithms accordingly to steer at-risk users toward more moderate groups. To test this hypothesis, we are investigating whether we can predict the probability of subsequent QAnon membership from pre-Q reddit posts in other (non QAnon related) subreddits.

CS230: Deep Learning, Fall 2022, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

In much of the existing NLP work on political extremism, the motivation is to automate detection or flagging of problematic language (e.g., political extremism, hate speech, suicidal ideation) [13] [14] [7]. In other words, because it is not feasible for human moderation teams to identify all extreme content manually, research has focused on machine learning methods to identify candidate extreme content in an initial pass. The algorithms employed for such detection range from traditional NLP techniques such as Latent Dirichlet Allocation [13] to classifiers such as SVM and Random Forest [7] to deep learning models [9]. A meta-analysis investigating 41 papers on hate speech detection concluded that SVM has been the most commonly used algorithm, though deep learning and hybrid methods have been gaining popularity recently [14].

While reactive detection of political extremism is useful for content moderation, it is also important to predict extremism before it results in real-world harm. For example, Habib et al. argue that reddit's current strategy of banning or quarantining communities only after real-world harm is insufficient and use Logistic Regression and Random Forest models to predict subreddits' probabilities of becoming problematic [8]. Other work similarly predicts extremist outcomes at the community [6] or topic level [13] using largely standard machine learning classifiers. While these units of prediction make sense insofar as content moderation often happens at the community level (e.g., banning an entire subreddit), intervention could also occur at the individual level. There is little existing work on individual-level prediction, and that which does exist relies on traditional machine learning classifiers such as Logistic Regression and Random Forest, using handcrafted numeric features such as mean tweets per day and mean retweets [5]. While these approaches achieve reasonable performance, they omit features encoded in the language itself and require manual feature engineering. Language models in deep learning address both these issues; hence, the novelty of the current work is predicting individual-level development of extremist attitudes using deep learning language models.

3 Dataset and features

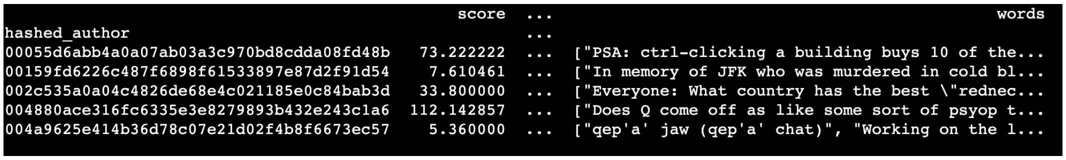
To construct our positive class dataset (QAnon extremists), we use two main datasets from [4], available here. Specifically we use HashedQSubmissionsRawCombined.csv as our feature set, which queried the Pushshift API (psaw) to fetch 2099875 unique posts from 13182 unique Q-users starting one year before the first Q-drop. We join this csv to HashedallAuthorStatus.csv, which labels these Q-users as QAnon-enthusiastic (3506 users; defined as top 25 percent most active Q-users, authors who have published five or more submissions in the 19 QAnon-focused subreddits) or QAnon-interested (9676 users; defined as the remaining users who did not meet the enthusiastic criteria).

One challenge or limitation of the Engels et al. dataset is that it is biased toward users affiliated with QAnon in some way. Specifically, all users in the dataset were active in at least 1 of 19 QAnon-focused subreddits that reddit banned in September 2018 for policy violations. In addition, the label definition was handcrafted based on heuristic criteria. Indeed, upon manual inspection of a few posts we determined that the classification task of enthusiastic vs. interested users was not particularly well justified, and a better test of our research question would be to predict membership in QAnon forums. To do this, we assigned a label of 1 to all users in the Engel et al. dataset and augmented with negative class labels using the procedure described below.

We used the reddit Pushshift dataset to sample our negative dataset here. This is a public dataset that contains all reddit posts since 2006. Due to large data volume, we were only able to process all posts in January 2017, which falls in the middle of our positive class time frame. This contained 1.96M users with 11.3M posts. In order to preprocess our dataset in a performant and flexible way, we ingested both the Pushshift and QAnon dataset into a single PostgreSQL table, reconciling the data schema where possible. Any users who did not appear in the QAnon dataset were labeled 0. We exported the post processed data to csv, which our training program consumed. The rows were grouped by author, then post id, and sorted ascending by the creation timestamp of the post.

4 Method

Data preparation



```
                                               score  ...                                              words
hashed_author                                         ...
00055d6abb4a0a07ab03a3c970bd8cdda08fd48b   73.222222  ...  ["PSA: ctrl-clicking a building buys 10 of the...
00159fd6226c487f6898f61533897e87d2f91d54    7.610461  ...  ["In memory of JFK who was murdered in cold bl...
002c535a0a04c4826de68e4c021185e0c84bab3d   33.800000  ...  ["Everyone: What country has the best \"rednec...
004880ace316fc6335e3e8279893b432e243c1a6  112.142857  ...  ["Does Q come off as like some sort of psyop t...
004a9625e414b36d78c07e21d02f4b8f6673ec57    5.360000  ...  ["qep'a' jaw (qep'a' chat)", "Working on the l...
```

Figure 1: Training examples

Windowing
Since we are interested in language prior to QAnon membership, we determined the timestamp of each user's first post in one of the 19 identified QAnon affiliated subreddits. We then used this timestamp to filter out all posts for this user on or after this time from our dataset. This excluded about 50% of our QAnon dataset as some Q users' first post was in a QAnon affiliated one.

Filtering
While manually examining our data, we found a portion of the most prolific users producing more posts than possible for a human (e.g. >10k posts a month). Most of these posts have no upvotes or no comments. To weed out these likely bot users, we filtered out all authors who have a score of 1 or less (which means no upvotes) or posts with 0 comments. For our negative dataset, we further filtered down to users who posted between 20 and 1000 times (1000 being the upper limit of what is possible for a human to produce in a month, and 20 being the lower bound of the minimum data size we'd like per author). We then randomly sampled the same number of authors as our positive dataset.

Aggregation and join
Since we are making predictions at the user level, we group the raw submissions data by user and concatenate all posts from that user into a single array. Mean post count was 61.77 (SD=335.85) overall, 48.29 (SD=61.76) for negative class, and 75.24 (SD=470.53) for positive class. Mean words per post was 34.90 (SD=178.93) overall, 36.59 (SD=209.19) for positive class, and 32.39 (SD=120.391) for negative class. The total number of posts was 750945. We then join this user-level raw feature input to the label data. This leaves us with a total of 12370 users (6185 positive class, 6185 negative class). At this point we split into a 60/20/20 train/dev/test split given the small size of our dataset. 5 examples are shown in Figure 1.

Embeddings algorithms

Task-specific word embeddings
Using the concatenated post string per user, we first perform some text standardization (casting to lowercase, removing stopwords, removing punctuation, etc.). We then use a TextVectorization layer to map each word to an integer. Next, we train an embedding layer of length 512 for each word. To generate the input to the MLP baseline model, we then average over the word dimension to generate a 512-length vector for each user. For the RNN and LSTM models, we input the sequence of word embeddings.

Pre-trained sentence embeddings
There are two major limitations of the word embeddings described above. First, they are trained only on the present dataset rather than harnessing the benefit of large pre-trained language models. Second, they do not leverage the fact that words are grouped into posts. To address these limitations, our second embeddings approach uses pre-trained SBERT (Sentence BERT), treating each post as a sentence. Several versions of SBERT exist, but we used all-MiniLM-L6-v2 based on documentation suggesting that of general purpose SBERT models, it is 5 times faster and still offers good quality SBERT docs. Since SBERT has quadratic increasing memory and time consumption, we wanted to avoid encoding an entire post as they can be up to 10k words. We'd also like to capture the nature of the sentence sequences per post. Specifically, to generate the input to the MLP baseline model, we converted each post to a sequence of 150 word chunks, which was then passed through SBERT to generate a 384-length post embedding. We then average over the post dimension to generate a 384-length vector for each user. For the RNN and LSTM models, we used the same procedure but omitted averaging, instead using the sequence of embeddings as the input layer.

Model architectures

A diagram of each model architecture is available in the Appendix.

MLP

Our first baseline model is a simple feedforward neural network with 2 dense hidden layers and a single unit output layer. We use log loss as our cost function and the Adam optimizer with default hyperparameter settings. At each layer we reduce dimensionality, with 256 units at the first hidden layer, and 128 units at the second hidden layer. We use ReLU activation at each hidden layer. In an attempt to mitigate overfitting, we use L2 regularization at each hidden layer followed by batchnorm, as well as early stopping (num epochs=30, patience=3). We apply a sigmoid transform to the output to convert our predictions to probability space.

RNN

Due to the sequential structure of the post data, we built a many to one RNN model as a second baseline. The input is a sequence of trained word or BERT sentence embeddings for the author's entire post history, which we fed to a simple RNN layer with 64 hidden units. As with the MLP, the RNN applies a sigmoid activation on the output layer. It also uses the same cost function, Adam optimizer settings, and regularization strategies.

LSTM

Hoping to capture more long-term temporal dependencies, we built a many to one LSTM model. For the trained word embeddings, we used 2 hidden LSTM layers, one with 32 hidden units outputting a sequence and one with 32 hidden units. For the sentence embeddings, we also used 2 hidden LSTM layers. We took the ordered sentence chunk encodings per post and passed each of them to a LSTM with 32 hidden units. We then took the output of each of these LSTMs and combined them into a new sequence that we then passed to a second LSTM also with 32 hidden units. We used the same sigmoid activation on the output layer, cost function, Adam optimizer, and regularization strategies as the baseline models.

Hyperparameter tuning

We ran each {embedding, model architecture} combination over a grid search of the following hyperparameter values: 1) l2 penalty weight: [.001, .01, .1], 2) learning rate: [.0001, .001], 3) batch size: [16, 32], 4) dropout rate: [0.1, 0.25] . The results presented below represent the highest performing hyperparameter combination for each respective model.

Results

We used F1-score on the test set as our primary metric to compare model performance and also examined accuracy (appropriate in this case as our data augmentation procedure guaranteed balanced classes), precision, and recall. Results on the test set across all 6 models are shown in Table 1, and plots of train and dev set performance across epochs are available in the Appendix.

We see that each model architecture with sentence embeddings outperforms its word embeddings counterpart on F1-score. In addition, the advantage of sentence embeddings is stronger for the sequence models (RNN and LSTM) than the MLP baseline–while F1-score only increases from 0.61 to 0.62 in the MLP, it increases from 0.55 to 0.69 in the RNN and from 0.66 to 0.80 in the LSTM. The ancillary metrics also do not consistently improve with sentence embeddings for the MLP (e.g., precision drops from 0.89 to 0.47), whereas they universally improve with sentence embeddings for the sequence models. We also observe far less overfitting with sentence embeddings as seen by smaller metric divergence between train and dev performance across epochs. The advantage of sentence embeddings generally could be attributed to the fact that each embedding aligns with a natural semantic unit of a post. It could also be due to the use of a pre-trained large language model which introduces inductive bias and has a regularizing effect.

Performance also generally increases with model complexity, with LSTM outperforming the two baselines both with word embeddings and sentence embeddings. This is not surprising given the sequential nature of the post data, as well as its length. We would expect the sequence models to outperform the MLP as they can take advantage of word and post ordering to predict the user's end state. However, the advantage of the LSTM over the RNN is likely due to how long the input is. By assigning weights to more important information in the posts, LSTM mitigates the risk of vanishing gradients we might encounter by applying a vanilla RNN to long text sequences.

Embeddings	Model architecture	F1-Score	Accuracy	Precision	Recall
Word	MLP	0.61	0.75	0.89	0.49
Word	RNN	0.55	0.63	0.59	0.53
Word	LSTM	0.66	0.67	0.65	0.74
Sentence	MLP	0.62	0.47	0.47	0.97
Sentence	RNN	0.69	0.80	1.0	0.55
Sentence	LSTM	0.80	0.79	0.81	0.81

Table 1: Evaluation metrics

5 Conclusion/Future work

This research explored the viability of predicting the probability of subsequent membership in QAnon subreddits from the language in a user's preceding posts. While there is some prior work in the NLP space related to classification of extremist attitudes, there is little predictive research particularly at the individual level. What work does exist relies on traditional machine learning algorithms which cannot natively ingest text sequences and hence require manual feature engineering to convert to a tabular format. Therefore, we investigated whether we could leverage language embeddings and sequence models to automate the user-level prediction problem.

To do so, we explored two types of language embeddings (1. word embeddings trained as part of the current classification task 2. pre-trained sentence embeddings) applied to three model architectures (1. MLP, 2. RNN, 3. LSTM). We found that even the simplest baseline approach of averaging word embeddings produces above-chance performance with 75% accuracy on balanced classes and 0.61 F1-score, though we see evidence of overfitting. We mitigate overfitting and see significantly stronger performance by using pre-trained sentence embeddings (treating a post as a sentence) in sequence models, particularly LSTM which is well suited for long sequences, resulting in 79% accuracy and 0.80 F1-score.

There are several limitations of the current study which could be addressed in future work. First, our candidate embeddings vary on two dimensions: whether they were pre-trained, and the embedding unit (word vs. sentence). Therefore, it is difficult to identify which dimension is responsible for the increase in performance we observe with sentence embeddings. We could isolate this effect by testing pre-trained word embeddings, such as word2vec or GloVe, as well as task-specific sentence embeddings. Second, it is possible that the advantage of LSTM has more to do with the weights it assigns than its sequential nature–while we have some evidence of this from the RNN performance (which is poorer than LSTM), we could isolate this effect more rigorously by learning weights for different values in the embedding and applying them in the MLP. Third, there are likely other features besides post language that could be important predictors, such as images embedded in the posts, number of comments, user demographics, etc. We did not include such features in the current study but could explore concatenating language embeddings with these features in future work. Finally, this task suffers from a censoring problem–it is unknown whether the users labeled as a negative class may yet join an extremist group in the future. We could consider survival analysis techniques to address this concern.

In sum, we see that it is possible to predict future extremist online group membership from a user's language alone. Moreover, this effect cannot be attributed solely to the presence or absence of particular words–as we see superior performance with sentence embeddings and sequence models–suggesting that the evolution of extremist attitudes is encoded in the particular sequence of a user's post history itself. These findings demonstrate that content moderation teams and other areas of online community platforms can a priori identify users at risk of joining extremist groups and potentially causing real-world harm, and intervene to prevent such actions.

6 Code

https://github.com/isvezich/cs230-political-extremism

7 Contributions

Lillian: RNN, LSTM, refactoring
Stephanie: MLP baseline model, sentence embeddings
Both: data gathering and preprocessing, wrote final report

References

[1] Rob Brotherton. *Suspicious minds: Why we believe conspiracy theories*. Bloomsbury Publishing, 2015.

[2] Karen M Douglas, Robbie M Sutton, and Aleksandra Cichocka. The psychology of conspiracy theories. *Current directions in psychological science*, 26(6):538–542, 2017.

[3] Karen M Douglas, Joseph E Uscinski, Robbie M Sutton, Aleksandra Cichocka, Turkay Nefes, Chee Siang Ang, and Farzin Deravi. Understanding conspiracy theories. *Political Psychology*, 40:3–35, 2019.

[4] Kristen Engel, Yiqing Hua, Taixiang Zeng, and Mor Naaman. Characterizing reddit participation of users who engage in the qanon conspiracy theories. *Proceedings of the ACM on Human-Computer Interaction*, 6(CSCW1):1–22, 2022.

[5] Emilio Ferrara, Wen-Qiang Wang, Onur Varol, Alessandro Flammini, and Aram Galstyan. Predicting online extremism, content adopters, and interaction reciprocity. In *International conference on social informatics*, pages 22–39. Springer, 2016.

[6] Ted Grover and Gloria Mark. Detecting potential warning behaviors of ideological radicalization in an alt-right subreddit. *Proceedings of the International AAAI Conference on Web and Social Media*, 13(01):193–204, Jul. 2019.

[7] Prabhakar Gupta, Pulkit Varshney, and MPS Bhatia. Identifying radical social media posts using machine learning. Technical report, Tech. Rep., 2017, doi: 10.13140/RG. 2.2. 15311.53926, 2017.

[8] Hussam Habib, Maaz Bin Musa, Muhammad Fareed Zaffar, and Rishab Nithyanand. Are proactive interventions for reddit communities feasible? In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 264–274, 2022.

[9] Shynar Mussiraliyeva, Batyrkhan Omarov, Milana Bolatbek, Ruslan Ospanov, Gulshat Baispay, Zhanar Medetbek, and Zhastay Yeltay. Applying deep learning for extremism detection. In *International Conference on Advanced Informatics for Computing Research*, pages 597–605. Springer, 2020.

[10] Mallory Newall. More than 1 in 3 americans believe a 'deep state'is working to undermine trump. *Ipsos. Accessed*, 2(27):2022, 2020.

[11] Federal Bureau of Investigation. Adherence to qanon conspiracy theory by some domestic violent extremists. 2021.

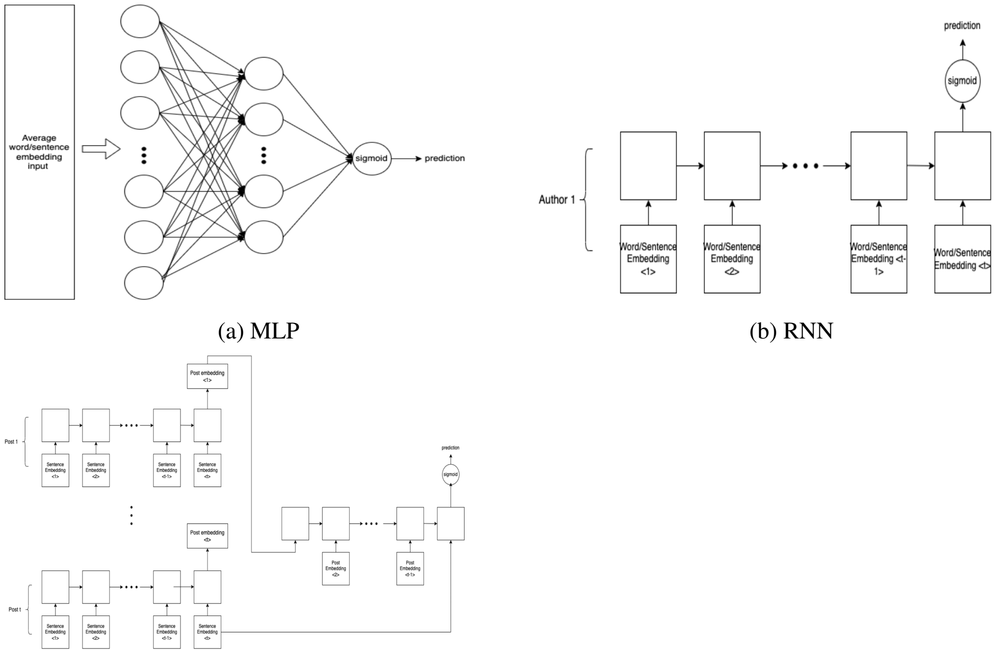
[12] Kevin Roose. What is qanon, the viral pro-trump conspiracy theory. *The New York Times*, 3, 2021.

[13] Richard F Sear et al. Dynamic latent dirichlet allocation tracks evolution of online hate topics. advances in artificial intelligence and machine learning. 2022; 2 (1): 17, 2010.

[14] Hyellamada Simon, Benson Yusuf Baha, and Etemi Joshua Garba. Trends in machine learning on automatic detection of hate speech on social media platforms: A systematic review. *FUW Trends in Science & Technology Journal*, 7(1):001–016, 2022.

[15] Brandy Zadrozny and Ben Collins. How three conspiracy theorists took 'q'and sparked qanon. *NBC News*, 14, 2018.

8 Appendix

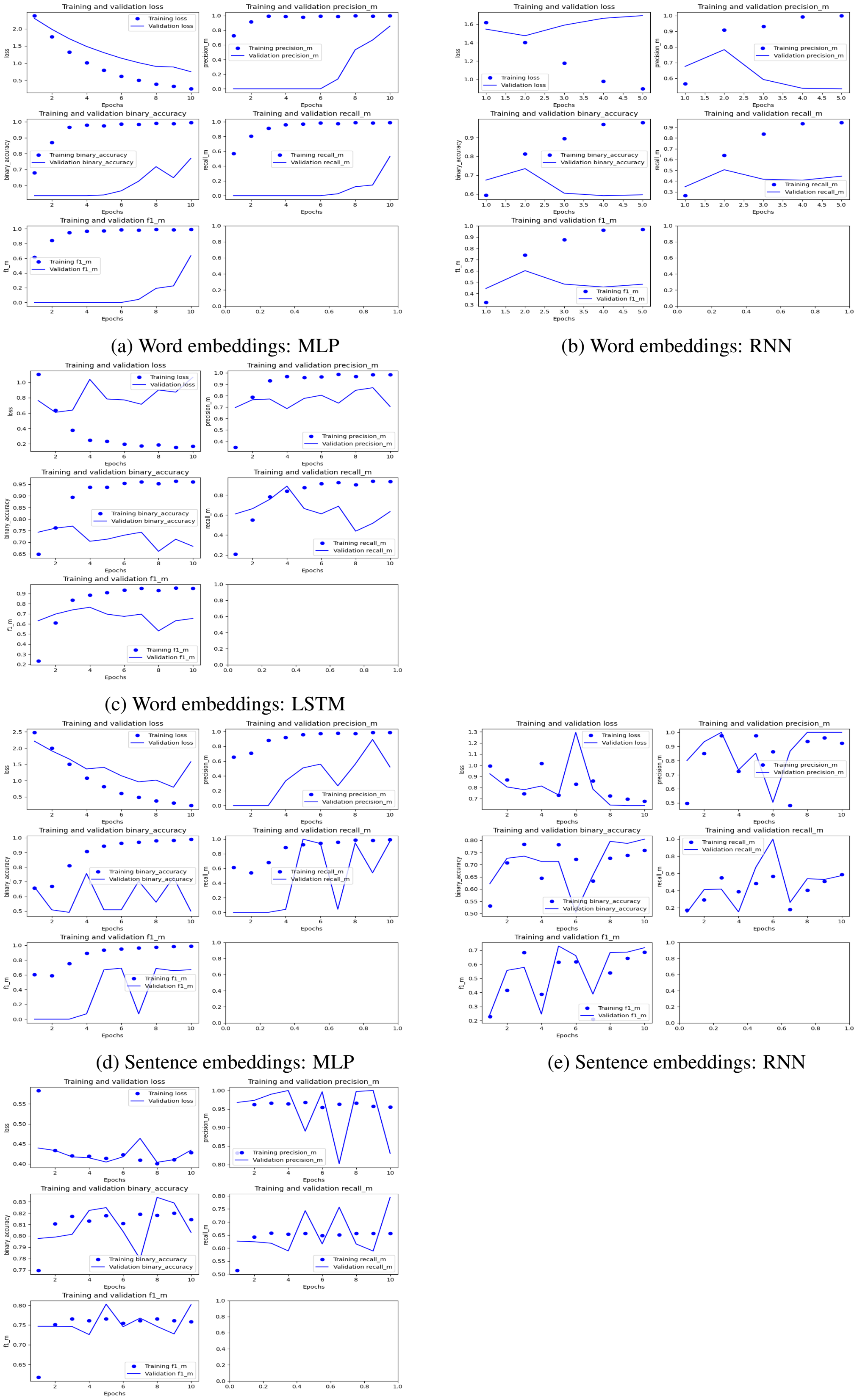


(a) MLP

(b) RNN

(c) LSTM

Figure 2: Model architecture diagrams



(a) Word embeddings: MLP

(b) Word embeddings: RNN

(c) Word embeddings: LSTM

(d) Sentence embeddings: MLP

(e) Sentence embeddings: RNN

(f) Sentence embeddings: LSTM

Figure 3: Evaluation metrics (Note that due to early stopping, different models have different number of epochs.)