Predicting Twitter Engagement With Deep Language Models

Maksims Volkovs
Layer 6 AI
maks@layer6.ai

Zhaoyue Cheng
Layer 6 AI
joey@layer6.ai

Mathieu Ravaut
Layer 6 AI
mathieu@layer6.ai

Hojin Yang
University of Toronto, Layer 6 AI
hojin@mie.utoronto.ca

Kevin Shen
Layer 6 AI
kevin@layer6.ai

Jin Peng Zhou
University of Toronto, Layer 6 AI
jin@layer6.ai

Anson Wong
Layer 6 AI
anson@layer6.ai

Saba Zuberi
Layer 6 AI
saba@layer6.ai

Ivan Zhang
Cohere
ivan@cohere.ai

Nick Frosst
Cohere
nick@cohere.ai

Helen Ngo
Cohere
helen@cohere.ai

Carol Chen
Cohere
carol@cohere.ai

Bharat Venkitesh
Cohere
bharat@cohere.ai

Stephen Gou
Cohere
stephen@cohere.ai

Aidan N. Gomez
Cohere
aidan@cohere.ai

ABSTRACT

Twitter has become one of the main information sharing platforms for millions of users world-wide. Numerous tweets are created daily, many with highly time sensitive content such as breaking news, new multimedia content or personal updates. Consequently, accurately recommending relevant tweets to users in a timely manner is a highly important and challenging problem. The 2020 ACM RecSys Challenge is aimed at benchmarking leading recommendation models for this task. The challenge is based on a large and recent dataset of over 200M tweet engagements released by Twitter with content in over 50 languages. In this work we present our approach where we leverage recent advances in deep language modeling and attention architectures, to combine information from extracted features, user engagement history and target tweet content. We first finetune leading multilingual language models M-BERT and XLM-R for Twitter data. Embeddings from these models are used to extract tweet and user history representations. We then combine all components together and jointly train them to maximize engagement prediction accuracy. Our approach achieves highly competitive performance placing 2'nd on the final private leaderboard. Full code is available here: https://github.com/layer6ai-labs/RecSys2020.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Neural networks**.

RecSysChallenge '20, September 26, 2020, Virtual Event

ACM ISBN 978-1-4503-8835-1/20/09.
https://doi.org/10.1145/3415959.3416000

KEYWORDS

Recommender Systems, Deep Learning, Language Modeling, Attention

ACM Reference Format:
Maksims Volkovs, Zhaoyue Cheng, Mathieu Ravaut, Hojin Yang, Kevin Shen, Jin Peng Zhou, Anson Wong, Saba Zuberi, Ivan Zhang, Nick Frosst, Helen Ngo, Carol Chen, Bharat Venkitesh, Stephen Gou, and Aidan N. Gomez. 2020. Predicting Twitter Engagement With Deep Language Models. In *Proceedings of the Recommender Systems Challenge 2020 (RecSysChallenge '20), September 26, 2020, Virtual Event*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3415959.3416000

1 INTRODUCTION

With over 300 million monthly active users, Twitter is one of the primary sources of real-time information around the world. Millions of tweets are created every day, and majority have time-sensitive content such as breaking news, new media releases or personal updates. To deal with such volume of information Twitter algorithmically selects relevant tweets to show in users' feeds. The recommendation system at the core of this process thus plays a critical role, and has to accurately infer relevance from limited available information. The *2020 ACM RecSys Challenge* is aimed at benchmarking recommender models for this task in a standardized setting. This challenge is based on a recent and large-scale dataset released by Twitter with over 200M engagements from 30M users and 90M tweets. There are four main types of engagements - *Retweet*, *Reply*, *Like* and *Comment*, and the goal of the challenge is to predict whether user will engage with a given tweet. Training engagements sampled from one week are released for model development, and engagements from the following week are used for evaluation. All evaluation is done by submitting model predictions to evaluation server, and test data is partitioned into public and private leaderboards. Throughout the challenge model scores

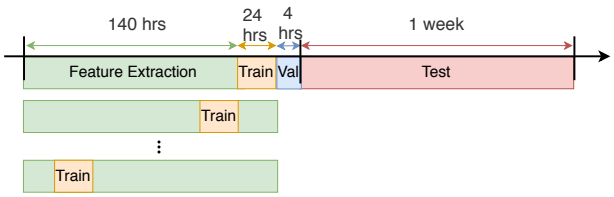


Figure 1: Data partitioning. We use a 24 hour sliding window approach to create the training set. To maximize training data all models are validated on a small 4 hour validation set forward in time. Public and private leaderboard test data is taken from the following week.

on the public leaderboard are visible to participants, while private leaderboard is released at the end and used for final team ranking.

The data for this challenge contains information on engaging users, tweets and tweet creators [1]. Particular attention is given to tweet content, and tokenized wordpiece ids from multilingual BERT (M-BERT) model [4] are provided for each tweet. The dataset is highly multinational with tweets in over 50 different languages. Moreover, all test tweets are cold start and have no engagement history. This simulates real-life application of the recommendation model that has to predict engagement for newly published tweets. In the cold-start setting tweet content becomes particularly important as it is one of the only sources of information available during prediction. Our approach focuses on this aspect and combines recent advances in deep language modeling with collaborative filtering in a neural network architecture that is trained jointly to maximize engagement prediction accuracy. We achieve strong performance placing second on the final private leaderboard out of over 30 teams.

2 APPROACH

2.1 Data Partitioning

To partition the data we first order all engagements by time. Negative user-tweet pairs where no engagement occurred are ordered based on tweet creation date. We then partition the ordered engagements forward in time to preserve temporal dynamics in the data. Specifically, last four hours of the seven day training period are used for model validation and the rest for training. Given the large data size, four hour period contains over 3.6M engagements, and we find that accuracy on this set generalises well to the public leaderboard. Training data is generated by applying a non-overlapping 24 hour sliding window to the training period. Engagements in the 24 hour window are taken as training targets and the rest are used for feature extraction. Note that we break temporal structure here and also extract features from engagements that are forward in time from the training window as shown in Figure 1. Empirically, we observe a considerable improvement in performance from applying this procedure as it significantly increases the amount of data available for feature extraction. For all experiments we use five consecutive training windows (going backwards from the start of the validation set) which results in a training set with 109,418,789 examples.

2.2 Model Architecture

The model architecture diagram of our approach is shown in Figure 2. Our model is based on a deep learning pipeline with three sources of input: 1) extracted features that describe engaging user, tweet creator and tweet content 2) language model embedding for tweet content 3) engagement history embeddings for the engaging user. These inputs are combined with attention followed by feed-forward neural network to generate 4-way predictions for each engagement type. Below we describe the main components of this architecture.

Feature Extraction We conduct extensive feature engineering to describe engaging user, tweet creator and target tweet, with particular focus on historical engagements and similarity between the three entities. In total we extract 467 features that can be partitioned into four main groups:

- *Engaging User* These features summarize engaging user's history on Twitter and current account status including created and engaged with tweets (by engagement type), follower and following count trends, account duration, verified status etc. Time-based features such as time since last engagement are particularly important here, and we observe significant boost in accuracy after adding them into the model.
- *Tweet Creator* Analogous to engaging user, we also summarize past history for tweet creator. In particular, we focus on previous tweets created by this user and level of engagement they received (popular or not).
- *Tweet Content* For target tweet we include all available content information including tweet type (retweet, reply etc.) text tokens, hashtags, media and language. Since all test tweets are cold start we do not use historical engagement information here. Tweet content features are further supplemented with embedding from the language model which we describe in the following section.
- *Interactions* As commonly done in collaborative filtering, we extract user-tweet and user-creator features to capture similarity between these entities. For user-tweet we compute content similarity between tweets that user has previously created or engaged with and target tweet. Similarly, for user-creator we focus on their joint engagement history, and summarize whether user has previously interacted with tweets from this creator and vice versa. We also extract collaborative filtering similarity between user and creator based on the binary interaction matrix $\mathbf{R}$ where $\mathbf{R}_{uu'} = 1$ if user u has previously engaged with any tweet from creator u'. Similarity between two rows/columns of $\mathbf{R}$ indicates the degree to which the corresponding users have similar engagement patterns for engaged/created tweets.

To make these features suitable for neural network training, we use 1-hot encoding for all categorical features, and apply the Yeo-Johnson [11] power transformation to all numeric features.

Language Model Tweet content features described above use 1-hot representations for the most commonly appearing wordpiece ids from tweet text. These features provide high level information about the tweet such as general topics and theme. However, they fail to capture more nuanced semantic structure and sentiment, that play an important role in users' engagement [5]. To extract

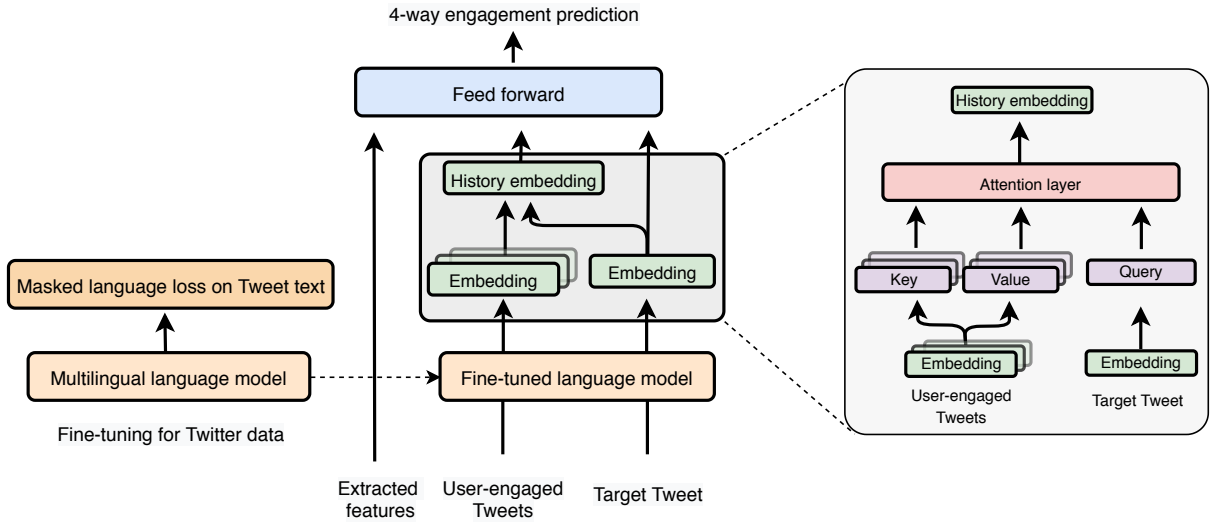


Figure 2: Model Architecture. Multilingual language model (M-BERT, XLM-R) is first fine-tuned with Twitter data using unsupervised masked language loss. The fine-tuned model is then used to extract embeddings for target tweet and most recent historical tweets that target user engaged with. Tweet embeddings are aggregated with attention and combined with extracted features in a feed forward network to output 4-way engagement prediction.

this information we leverage leading deep language models. Neural language modeling has seen tremendous improvement over the last few years, especially after the introduction of the Transformer architecture [10] and BERT [4]. Language models based on the Transformer [2, 6, 7] and pre-trained on Internet-scale data have pushed the state-of-the-art and surpassed human performance on many text comprehension tasks.

The majority of published language models are pre-trained on large text corpora such as Wikipedia or CommonCrawl, that typically contain longer and properly worded pieces of text. Tweets on the other hand are restricted to 280 characters, and users frequently use abbreviations and slang to fit into that requirement. To align the distribution of the language model with Twitter we apply further unsupervised fine-tuning with a masked language loss on tweet text. As the dataset is highly multilingual with tweets in over 50 languages, we use multilingual versions of BERT (M-BERT) [4] and XLM-R [3] models. Starting with the pre-trained models taken from the huggingface library [1][2], we use all unique training tweets to fine-tune the models and validation tweets that don't appear in training to evaluate perplexity. Figure 3a shows validation perplexities for both models, and we see there is a significant reduction in perplexity for both models. Throughout our experiments we also observe that language models fine-tuned for Twitter perform better on the downstream task of engagement prediction, so matching language distributions is important here. After fine-tuning, we use embeddings from the last layer of the language model as additional feature input for each tweet as shown in Figure 2.

User History with Attention Item similarity between past interactions and target item is frequently used in collaborative filtering as a robust way to estimate relevance [9]. In interaction features we leverage this approach and compute statistics on content similarity between tweets that user has engaged wit/created, and target tweet. However, these statistics are aggregated across the entire user history, and the model has no way of focusing on specific tweets that can be particularly useful for engagement prediction. We thus expand historical features by incorporating Transformer attention applied to language model embeddings. Specifically, given the target tweet embedding $\mathbf{q} \in \mathbb{R}^{d \times 1}$ we compute attention between $\mathbf{q}$ and n historical tweets $\mathbf{V}_e \in \mathbb{R}^{d \times n}$ that the user has engaged with:

$$\text{Attn}(\mathbf{q}, \mathbf{V}_e) = \text{Softmax}\left(\frac{1}{\sqrt{d}}(\mathbf{W}_q\mathbf{q})^T\mathbf{W}_k\mathbf{V}_e\right)(\mathbf{W}_v\mathbf{V}_e)^T \quad (1)$$

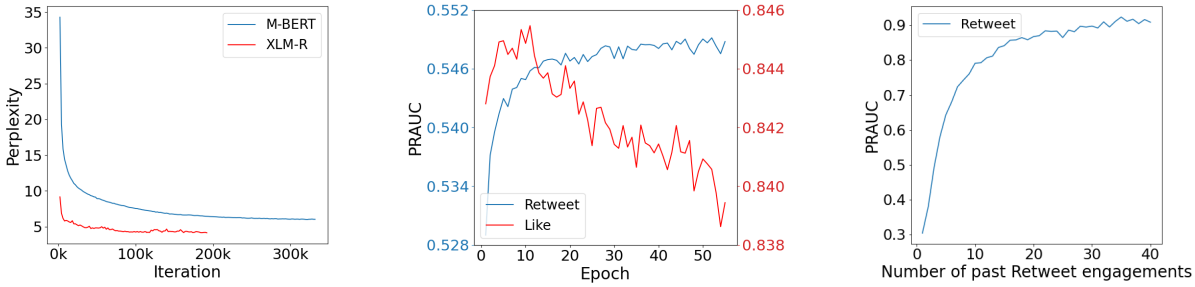
where $\mathbf{W}_q$, $\mathbf{W}_k$ and $\mathbf{W}_v \in \mathbb{R}^{d' \times d}$ are weight matrices to be learned. This operation outputs the attended target tweet embedding where Softmax weights determine how much importance is given to each historical engagement. We apply separate attention for each engagement type so $e \in \{Retweet, Reply, Like, Comment\}$, and resulting attended embeddings are concatenated together as additional input into the model. To reduce computational load we only consider (up to) the last $n = 10$ tweets for each engagement type.

Feed-Forward Classifier Extracted features, target tweet language model embedding and attended user history are first passed through individual feed-forward networks with one fully connected layer and ReLU activations. Outputs from each network are then concatenated together and passed through another feed-forward network to get engagement probabilities. This network has the following architecture $5000 \rightarrow 3000 \rightarrow 2000 \rightarrow 1000 \rightarrow 500 \rightarrow 4$ with ReLU activations in each layer. Since users can do multiple engagements on the same tweet, we apply sigmoid activations in the final layer, and optimize binary cross entropy for each engagement

[1] https://huggingface.co/bert-base-multilingual-cased
[2] https://huggingface.co/xlm-roberta-large

Table 1: Model Performance. Top half shows ablated model performance on the public leaderboard; bottom half shows final team ranking on the private leaderboard. We extract a total of 467 features and use them as base input for every model. BERT and XML-R embedding dimensions are 768 and 1024 respectively. History model uses up to 10 (if available) most recent tweets that the target user engaged with for each of the 4 engagement types. Our team *learner* achieves 2'nd place on the final private leaderboard.

Model	Input size	Average PRAUC	Average RCE	Retweet PRAUC	Retweet RCE	Reply PRAUC	Reply RCE	Like PRAUC	Like RCE	Comment PRAUC	Comment RCE
features + XGBoost	467	38.57	20.33	52.09	28.20	19.37	19.47	76.12	21.00	6.71	12.64
features + MLP	467	38.30	20.26	51.95	28.16	21.31	20.77	76.08	21.80	6.06	11.88
features + M-BERT	467+768	39.21	21.24	52.76	29.24	20.76	20.60	76.86	22.19	6.54	12.91
features + M-BERT + end-to-end	467+768	39.46	21.31	53.03	29.41	21.31	20.77	76.84	22.33	6.66	12.72
features + XLM-R	467+1024	39.48	21.71	53.24	29.86	21.20	20.88	76.71	22.89	6.76	13.20
features + XLM-R + history	467+1024+1024*4*10	39.74	21.74	53.18	29.28	21.36	21.05	77.51	23.49	6.91	13.14
Blend		**40.28**	**22.52**	**54.01**	**30.05**	**22.06**	**21.74**	**77.60**	**23.90**	**7.46**	**14.40**
Team name	**Rank**										
NVIDIA RAPIDS.AI	1	45.50	33.14	61.11	37.91	21.85	24.23	91.08	53.01	7.96	17.40
learner	**2**	40.33	23.23	53.95	30.96	22.18	21.94	77.61	25.42	7.57	14.61
Team Wantedly	3	39.06	22.53	52.66	30.06	19.18	20.44	77.16	24.76	7.24	14.86
learner_recsys	4	44.80	12.79	45.29	22.50	11.61	10.42	72.21	18.28	50.37	-0.04
BanaNeverAlone	5	39.15	18.87	50.42	27.21	18.50	18.71	75.31	21.20	12.37	8.34



(a) (b) (c)

Figure 3: a) Validation perplexity when M-BERT and XLM-R models are fine-tuned with a masked language loss on Twitter data. b) Validation PRAUC curves for Retweet and Like engagements, curves for Reply and Comment look similar to Retweet. c) Validation Retweet PRAUC grouped by number of historical Retweet engagements for each user.

type:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{e} y_{ie}\log\hat{y}_{ie} + (1-y_{ie})\log\hat{y}_{ie} \quad (2)$$

where N is the number of training examples, e is the engagement type, y_{ie} is 1 if user engaged with the tweet in the form of e and 0 otherwise, and $\hat{y}_{ie}$ is the predicted engagement probability. Note that all components of our architecture including language model embeddings are differentiable and can be trained end-to-end.

3 EXPERIMENTS

Training Details All experiments are conducted on Ubuntu servers with IBM POWER9 CPUs @ 3.8GHz, 600GB RAM, and NVIDIA Tesla V100 GPUs. Following the challenge rules [1], we use PRAUC and RCE metrics to evaluate model performances. M-BERT and XLM-R language models are fine-tuned with masked language loss on the training tweets until validation perplexity plateaus. We pre-process tweets by decoding wordpiece ids back to original text and replacing all mentions and urls with [MENTION] and [URL] tokens. This significantly reduces the number of uninformative rare tokens making the models more robust. We then re-tokenize the text using pre-trained tokenizer that comes with each model. M-BERT is fine-tuned with the AdamW optimizer [8], mixed precision, and a batch size of 256 (64 per GPU). Learning rate is warmed up for 10,000 iterations until 1e-4 with linear decay after. XLM-R is also fine-tuned with the AdamW optimizer and a batch size of 400 (50 per GPU). Learning rate is warmed up for 4,000 iterations until 5e-5 and then linearly decayed.

Figure 3a shows validation perplexity curves when M-BERT and XLM-R models are fine-tuned with a masked language loss on tweet text. We see that there is a significant reduction in perplexity for both models. This is expected since Twitter text distribution is significantly different from Wikipedia and related text corpuses on which these language models are pre-trained. We also see that XLM-R starts with a much lower perplexity than M-BERT, indicating that increasing model capacity is beneficial for language modeling task even when target word distribution is highly irregular. After unsupervised fine-tuning, we use the models to get tweet embeddings. Specifically, we average output token embeddings from the last hidden layer excluding the [CLS] token. For the end-to-end

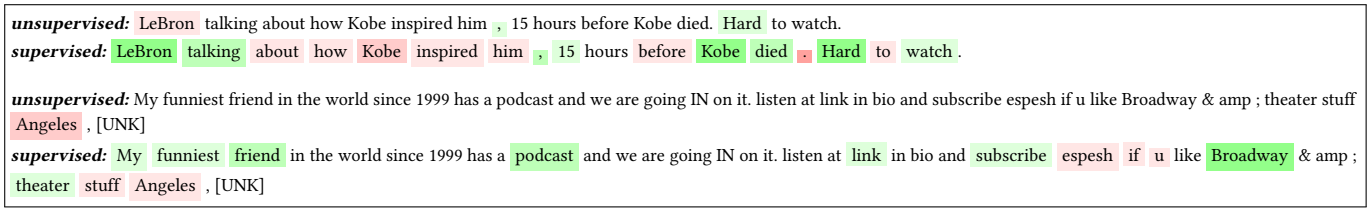


Figure 4: M-BERT Attention Visualization. Colors represent the change (green=increase, red=decrease) in attention weights relative to the off-the-shelve M-BERT model from huggingface. For each tweet, the first row corresponds to the model fine-tuned with masked language loss on tweet text. Second row is from the model that is additionally fine-tuned for supervised engagement prediction.

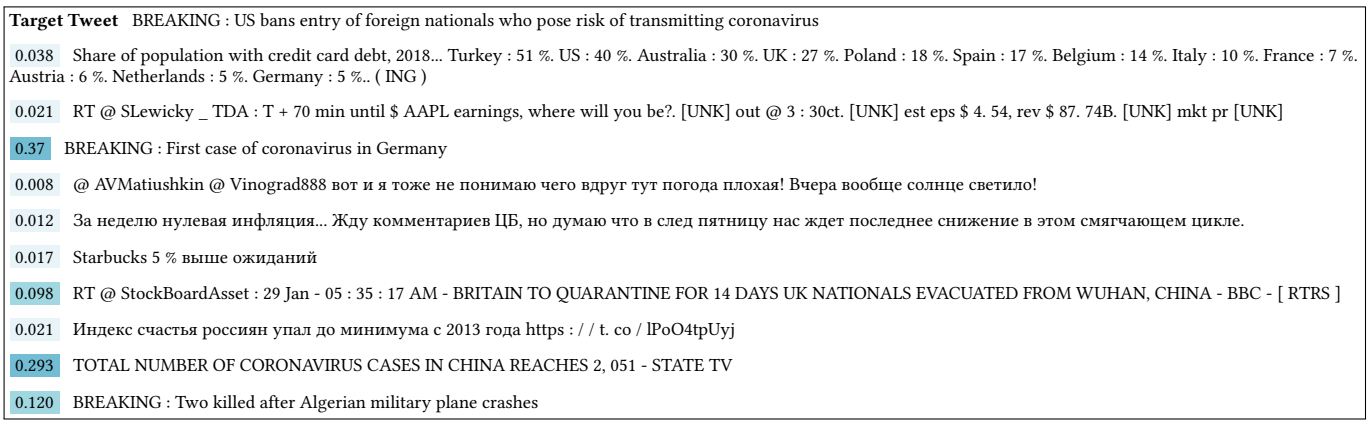


Figure 5: Attended User History Example. First row is the target tweet about coronavirus that the user liked. Subsequent rows are most recent ten tweets that this user previously liked with corresponding attention weights. Historical tweets are in both English and Russian, and the model is primarily focusing on tweets relevant to coronavirus.

experiments, we first train the feed-forward classifier only, keeping everything else fixed. We then further fine-tune the language model with the supervised loss, and finally train the entire architecture jointly. During joint training we use a smaller learning rate of 1e-5 for the language model and a larger one of 5e-5 for the classifier. In our experiments this is the only setting that produced improvements in performance, while others either overfitted or diverged.

Ablation Results Top half of Table 1 shows ablation performance on the public leaderboard. Starting with the 467 extracted features as base, we increase model complexity by first adding language model tweet embeddings and then attended user history. Form the table we see that adding language model embeddings, particularly from the larger XLM-R model, improves average PRAUC and RCE by over a point. Smaller gains are obtained from end-to-end training and attended user history. Due to runtime complexity and heavy memory requirements of these models we are only able to train several of them throughout the competition. Given promising early results, we believe that significant further gains can be obtained with proper parameter tuning and training to convergence. Bottom half of the table shows final team rankings on the private leaderboard. Our final submission uses a blend of several models and achieves 2'nd place on the private leaderboard. We also see that our model generalises well between the two leaderboards with comparable performance on both PRAUC and RCE metrics.

Figure 3b shows validation PRAUC curves for Retweet and Like engagements for the *features + MLP* model. We consistently observe that model performance on Like exhibits over-fitting early in training, while other three engagements continue to improve. To deal with this we use a different early snapshot of each model for Like predictions. Figure 3c shows validation Retweet PRAUC grouped by number of historical Retweet engagements for each user. As expected we see a strong pattern where more historical data leads to better performance. Improvement in performance is particularly significant for the first 10 engagements and starts to plateau after 20.

Qualitative Analysis Figure 4 shows the change in attention weights relative to the base M-BERT model loaded from huggingface. We show relative change when M-BERT is fine-tuned for Twitter data with a masked language loss (*unsupervised*), and when the same model is further fine-tuned for the supervised engagement prediction task (*supervised*). We see that supervised training leads to a larger relative change in attention than unsupervised. In particular, supervised training changes attention the most for entities such as *Lebron*, *Kobe* and *Broadway*, and parts containing sentiment such as *hard to watch* and *my funniest friend*.

Figure 5 shows visualization of the attended user history from the *features + XLM-R + attention* model. The target tweet is about coronavirus, and from history attention weights we see that the model is primarily focusing on coronavirus related tweets that user liked in the past. This example also demonstrates that recent advances in language modeling have enabled effective reasoning in multilingual domains. Historical tweets are in both English and Russian, however, tweets in Russian are not related to coronavirus and the model mostly ignores them.

4 CONCLUSION

In this paper we describe our approach to the 2020 ACM RecSys Challenge organized by Twitter. Our model combines deep language models with attention over historical engagements and extracted features to predict future engagements. We achieve highly competitive performance placing 2'nd on the final leaderboard out of over 30 teams. Future work involves additional investigation into deep learning architectures for this task.

REFERENCES

[1] Luca Belli, Sofia Ira Ktena, Alykhan Tejani, Alexandre Lung-Yut-Fon, Frank Portman, Xiao Zhu, Yuanpu Xie, Akshay Gupta, Michael Bronstein, Amra Delić, et al. 2020. Privacy-Preserving Recommender Systems Challenge on Twitter's Home Timeline. *arXiv preprint arXiv:2004.13715* (2020).

[2] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

[3] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. In *Association for Computational Linguistics*.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics*.

[5] Yuheng Hu, Shelly Farnham, and Kartik Talamadupula. 2015. Predicting user engagement on twitter with real-world events. In *Ninth International AAAI Conference on Web and Social Media*.

[6] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite Bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

[7] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[8] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[9] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *International World Wide Web Conference*.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neural Information Processing Systems*.

[11] In-Kwon Yeo and Richard A Johnson. 2000. A new family of power transformations to improve normality or symmetry. *Biometrika* 87, 4 (2000), 954–959.