Predictive Modeling Using Linguistic Signal for Suicidality in Social Media : Task B

Anchit Jain[1], Angelos Mavrogiannis [1], Divya Kothandaraman[1]
Sanket Doshi [3], Yashish Maduwantha[2],
[1] Department of Computer Science, University of Maryland,
[2] Department of Electrical and Computer Engineering, University of Maryland,
[3] Department of Physics, University of Maryland

1 Problem Statement

According to the World Health Organization, close to 800,000 people die every year due to suicide. [1] Suicide prevention is extremely important, but the assessment for suicide risk is hard and requires the patient to visit a clinician and that is only if a patient willingly decides to do so. Given posts from an anonymous social media platform like reddit, our goal is to automatically identify if a user is at risk for suicide. We are provided with access to posts from the subreddit 'SuicideWatch', which contains posts related to mental health, and posts from other subreddits, which may be indicative of the users' emotional state.

2 Data Pre-processing and Baselines

Data preprocessing, especially from a social media platform like reddit, can be quite tricky. The posts might be incoherent, including slang or colloquial English. Moreover, punctuation encapsulates emotions, and pronouns and conjunctions play an important role in the meaning of a sentence. Hence, there is a need to extract data in a way that does not miss out on important phrases that encode emotions.

Data pre-processing (Figure 1) We drop posts with nans in their post bodies and control user posts with nan labels. Basic feature engineering techniques involve using the bag of words (BoW) and the tf-idf feature representations. This involves n-gram generation, tokenization, and stopwords removal. We retain instances of self-references which might be crucial in the extraction of emotions.

Baseline experiments: We experiment with two baseline models: logistic regression (LR) and support vector machines (SVM). For the LR baseline, the default background pre-processing by sklearn CountVectorizer[2] For the SVM baseline, we have used Spacy for tokenization and lemmatization.We use "aggregate" BoW or tf-idf representations of the post bodies for each user. Test data corresponds to the test subset from the crowd annotated dataset. Logistic regression results can be found in Table 1. Though more data (using expert annotations in addition to crowd data for training) results in lower accuracy and F1 score, the FPR and FNR are lower which means there is a lower misclassification rate! Testing is on crowd annotated data due to differences in annotation schemes (in terms of expertise of the people labeling) potentially causing "domain shift". We suspect a dip in accuracy / F1 when expert data is also used in the training. SVM results in an accuracy and F1 score of 0.64 and 0.5, respectively.

[1] https://www.who.int/teams/mental-health-and-substance-use/suicide-data
[2] https://scikit-learn.org/stable/modules/generated/sklearn.feature$_extraction.text.CountVectorizer.html$

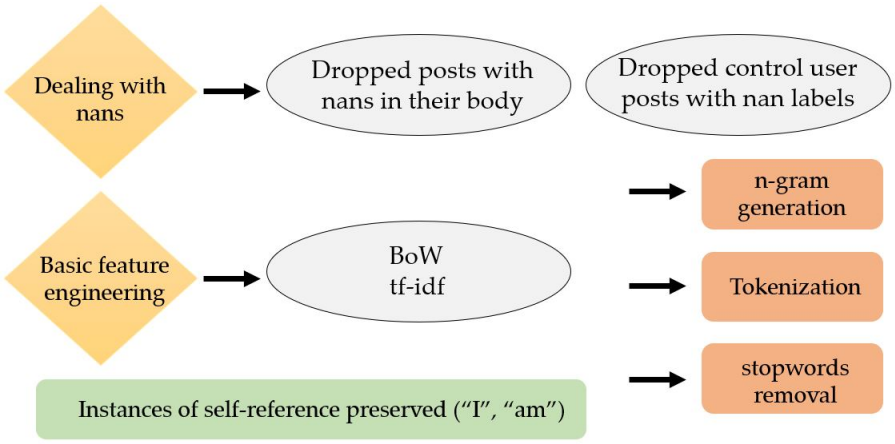

Figure 1: Methodology for data preprocessing

Training dataset	Crowd	Crowd + Expert
Accuracy	0.64	0.616
F1 score	0.5	0.4
FPR	0.219	0.164
FNR	0.4423	0.3
Conf. matrix	[[0.456 0.128][0.232 0.184]]	[[0.488 0.096][0.288 0.128]]

Table 1: Logistic regression baseline results.

3 Feature Engineering

3.1 Importance weighting

The baselines involved the usage of posts from all subreddits. We analyze (using Logistic Regression, crowd annotated data for training) if assigning a higher weight to SuicideWatch results in a better score in Table 2. Using only SuicideWatch posts yields the best results. The intuition is that SuicideWatch provides knowledge relevant to the task at hand, while we conjecture that other subreddits divert the attention.

3.2 Stopwords

Is it useful to remove English stopwords? Not really! Stopwords like "I", "me", help in capturing the mood of the user. Removing stopwords (and training using the baseline LR model, with crowd-annotated posts on only SuicideWatch posts) results in an accuracy and F1 score of 0.68 and 0.39, respectively. We believe a better extension would be to selectively preserve stopwords with certain POS tags, pronouns in this case.

3.3 Bigram models

Bigram models can capture context better than unigram models. The number of bigrams is quite large. However, it may not be optimal to use all features. Hence, a more reasonable solution is to pick the top n features. Ablation experiments reveal optimal performance at 50000 features. The

w	Accuracy	F1	Confusion matrix	FPR	FNR
1	0.64	0.5	[[0.456 0.128][0.232 0.184]]	0.219	0.4423
0.1	0.6	0.43	[[0.448 0.136][0.264 0.152]]	0.23	0.36
0	0.7	0.44	[[0.584 0.][0.296 0.12]]	0	0.288

Table 2: Does assigning a higher weight to SuicideWatch posts help? w denotes the weight to posts from subreddits other than SuicideWatch.

n	Accuracy	F1	Confusion matrix	FPR	FNR
Unigram (only)	0.7	0.44	[[0.584 0.][0.296 0.12]]	0	0.288
All features	0.728	0.527	[[0.576 0.008][0.264 0.152]]	0.01	0.365
10000	0.72	0.4927	[[0.584 0.][0.28 0.136]]	0	0.32
50000	0.74	0.52	[[0.584 0.][0.27 0.152]	0	0.36
100000	0.728	0.527	[[0.576 0.008][0.264 0.152]]	0.01	0.36

Table 3: Results using the bigram representation. n denotes that top n bigram+unigram features are used for training.

experiments (with LR baseline, SuicideWatch posts, crowd annotated data) can be found in Table 3. The top n features contain words that are most indicative of suicide risk like "depression", "hope", "worthless", "excitement", etc.

3.4 Word Embedding

We used a GloVe pre-trained model [4] to generate the word embeddings for the neural language model in section 3.9. We trained our model with 50-, 100-, 200- and 300-dimensional embeddings and noticed that the best classification metrics are achieved with the 50-dimensional embeddings so we used these in all of our experiments. We used GloVe over BERT due to the fact that we used a Hierarchical Attention Network (HAN) which explicitly introduces attention to capture contextual information. (But as a future experiment we would like to see how the BERT embeddings would perform when fed to a HAN implementation)

3.5 Emotion Features

Inspired by the work in [6], we use the NRC Word-Emotion Association Lexicon [3] to capture the emotional intensity of the posts. The emotions we consider in our analysis are {anger, anticipation, disgust, fear, joy, sadness, surprise, trust}. We begin by counting the total emotion-related tokens in each post, but realize that this metric could not differentiate between positive and negative posts. To this end, we make use of the emotional intensity scores for words provided by the NRC lexicon. The score for a word ranges from 0 to 1, depending on the amount of emotion that it conveys (0: lowest amount, 1: highest amount). The final emotion feature vector for a post consists of the (8) values that were accumulated by adding the scores for all respective emotion-related words of a post. A downside of this approach lies in the absence of ngrams of higher order to capture the context of a sentence more accurately. By only considering unigrams, sentences that include an emotionally intense word preceded by a negation, e.g. "I am not happy" will actually convey the exactly opposite meaning and emotion than the one intended, and therefore the emotion intensity score will not reflect the true emotion of the post. We try to offset this shortcoming with the addition of the GloVe embeddings which are inherently able to capture some context in sentences.

3.6 Mental Disease Lexicon

Hypothesizing that mentally ill users are more likely associated with a higher suicide risk, we follow the work of Zirikly et al. [8] in building a mental disease lexicon (mentalDisLex) which is an alphabetical list of mental disorders [3] Based on this lexicon, we construct a feature value for each post which is equal to the count of the number of terms from the lexicon that are detected in each post. We observe that posts with a higher score exhibit a high probability of suicide risk, as expected.

3.7 User-level Latent Dirichlet Allocation (LDA) on subreddit names

We believe that the titles of the subreddits associated with each user's set of posts encode semantic information that can help us segment the entire pool of users into meaningful clusters. To this end, we use Latent Dirichlet Allocation (LDA) [1] to infer a 10-topic model on the crowd training set. More specifically, we assign a bag of $\{s_1, s_2, \ldots, s_n\}$ to each user, where s_i represents the subreddit name for each of the user's n posts. For the LDA model, we treat these bags as documents, and the names of the subreddits as our vocabulary. We construct a 10-dimensional feature vector that

[3]https://mental-health-matters.com/psychological-disorders/alphabetical-list-of-mental-disorders/

Topic	Topic words
Mental-Health	[like, know, want, ve, get, feel, really, time, would, life]
Neutral (Gendered)	[women, men, like, food, would, female, one, time, get, male]
Reddit	[post, like, would, reddit, subreddit, one, please, new, people, want]
Music	[music, song, like, love, would, songs, one, know, album, show]
Mental-Health	[one, would, think, people, like, know, time, could, really, life]
Car/Motorsports	[car, san, prix, mph, get, weather, bike, road, new, one]
Drugs	[mg, like, ve, get, time, day, feel, would, prescription, really]
Games	[fire, damage, level, one, like, get, would, use, game, time]
Games	[game, get, use, ve, anyone, help, one, using, like, would]
Craigslist	[free, via, craigslist, ifttt, table, pick, good, wood, condition, new]
Mental Health	[would, get, looking, online, need, help, like, work, know, new]
Religion	[people, like, would, know, one, even, god, think, christian, ve]
Sports	[team, get, like, would, one, think, season, time, game, hair]
Neutral	[back, like, one, get, could, got, would, said, time, around]
Games (Pokemon)	[keys, trade, items, like, add, offers, want, would, pokemon, water]
Neutral	[week, ve, day, weight, dog, like, first, old, back, time]
Movies	[one, movie, time, day, watch, year, would, first, th, going"]
Cards (Game)	[amp, cards, buy, deck, card, good, looking, size, new, know]
Technology (Computer)	[gb, build, card, power, cpu, case, drive, price, video, memory]
Games (General)	[game, like, play, games, would, get, ve, really, looking, people]

Table 4: Results from our 20-topic model. We have interpreted Topic names and Topic words for corresponding topics

consists of the posterior probabilities for the derived topics. Based on the number of documents (496 users after removing control users), we believe that a reasonable range of topics for our training set is $\{5 - 10 - 15\}$. We find the best-performing number to be 10, but we further attempt to optimize for the α and β parameters while maximizing the best-performing coherence measure C_v, as presented in [5]. However, after performing a grid search over the number of topics, α, and β, the model with the highest coherence is outperformed by our initial 10-topic model, so we retain that for our analysis. [4]

3.8 Post-level Latent Dirichlet Allocation (LDA) on post title and body

In addition to the subreddit names, we also believe that post title combined with the post body encodes useful semantic information. So, we use a LDA model [1] to infer a 20-topic model on the crowd training dataset. Specifically, we concatenate the post titles with the post body to form a document for each post and use these documents to form a vocabulary. It must be noted that due to the sheer volume of posts, it was very time-consuming to process the whole vocabulary. As a result, we filter out words that occur in less than 15 documents and in more than 50% of the documents and keep the top 100000 words. After processing the vocabulary and training our 20 topic model, we extract 20-dimensional posterior probabilities for each document. Based on the number of documents (700,000 after removing posts with Nan bodies), we believe that the ideal number of topics is (10-15-20-25-30-35). From Figure 2, we can clearly see that the 20-topic model achieves the highest coherence score out of any other topic model. This 20-topic model achieves a perplexity of -8.52 which is the best perplexity out of 10,15,20,25,30,35 topic models which were trained on this dataset.

Table 4 shows the topic words and the topic names that our model produced. Examining the subreddit names and the posts made by the users, it is clear that a lot of users made gaming, sports, and technology related posts. These posts provide few, latent indicators for suicidality risk but other posts on suicidewatch provide stronger indicators for suicidality risk.

3.9 Timestamp features

We believe that the time of day and the day of the week when the post was made may contain important temporal information. The rationale being, a person may feel lonelier on a Friday compared to other days or in the night/evening compared to other times of the day.

[4] We used Gensim, https://radimrehurek.com/gensim/models/ldamulticore.html

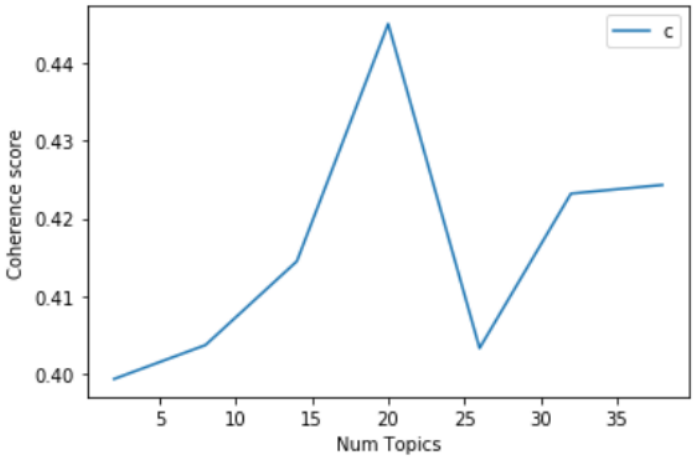

Figure 2: Coherence scores for each K-topic model (K is between 1 and 40)

In this spirit, we've converted the raw Unix timestamps from the data into date-time objects and utilized the hour and day of the week as features in our model. We have made the simplifying assumption of most users belonging to the United States timezone which justifies overlooking details pertaining to different time zones.

4 Hierarchical Attention Network (HAN) and Proposed Extension

In this section, we will review the original HAN architecture proposed by [7] and how it can be used for text classification. Then, we explain how we propose to extend/modify the architecture to address the suicide risk classification task at hand. We show how the architecture can be leveraged to assess the impact of the several features mentioned in the previous sections.

4.1 HAN: A Rapid-fire Introduction

The HAN architecture of [7] highlights the hierarchical structure of text documents in expressing a document as a weighted sum of sentence representations

$$v = \sum_i \alpha_i h_i \tag{1}$$

and sentences as weighted sums of word representations

$$s_i = \sum_t \alpha_{it} h_{it} \tag{2}$$

Where the $\alpha_{i(t)}$s are attention coefficients indicating how important each word is for a sentence and how important each sentence is for a document.

$$\begin{aligned} \alpha_i &= \frac{exp(u_i^T u_s)}{\sum_i exp(u_i^T u_s)} \\ \alpha_{it} &= \frac{exp(u_{it}^T u_w)}{\sum_i exp(u_{it}^T u_w)} \end{aligned} \tag{3}$$

u_s, u_w are context vectors that are learned simultaneously with other parameters and $u_{i(t)}$ are " improved annotations" derived from hidden representations/encodings of words and sentences as follows

$$\begin{aligned} u_i &= tanh(W_s h_i + b_s) \\ u_{it} &= tanh(W_w h_{it} + b_w) \end{aligned} \tag{4}$$

and

$$\begin{aligned} \vec{h_i} &= G\vec{R}U(s_i), \qquad \vec{h_{it}} = G\vec{R}U(W_e w_{it}) \\ \overleftarrow{h_i} &= G\overleftarrow{R}U(s_i), \qquad \overleftarrow{h_{it}} = G\overleftarrow{R}U(W_e w_{it}) \\ h_i &= [\vec{h_i}, \overleftarrow{h_i}], \qquad h_{it} = [\vec{h_{it}}, \overleftarrow{h_{it}}] \end{aligned} \tag{5}$$

are the respective annotations/hidden representations/encodings for sentences and words.

Let's consider an example of one of the Reddit post texts to see how a trained HAN picks important sentences and words within to pay attention to. One can extract the attention coefficients to find these and we do it with the help of the tutorial from [2].

We can take the text corresponding to "post_id: 2u4wgg". In which case, the summary (3 sentences with the highest weights) is: [['I attempted just a couple of days ago and am now trying to live with the guilt.', 'I hurt my friends who are like family to me.', 'They had to read goodbye letters, and now I feel *so* bad about doing that to them.'],

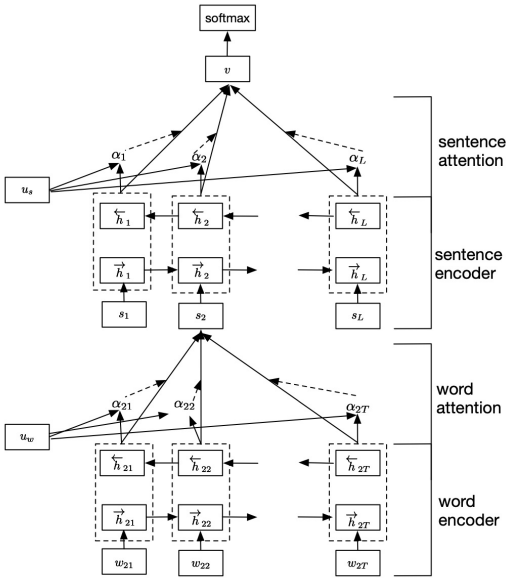

Figure 3: Original HAN architecture. Taken from [7]

4.2 Extended HAN, user-level, and post-level classification

For our task, the question we wish to address is "What aspects/features, out of all mentioned above are most important in predicting suicidality?" In order to leverage the posts data at hand, and to be able to make a user-level classification of high/low suicide risk, we propose an extension to the HAN architecture as shown in Fig. 4.

In this setup, the various feature sub-vectors (post body, post title LDA, timestamp features, etc.) serve as "words" to the "sentence" which is one post and the different posts aggregate to form a "document" which is the user to be classified into high/low risk of suicide. Due to implementation difficulties, the authors have resorted to a corresponding "post-level" architecture shown in Fig. 5.

In this setup, all the features are utilized via concatenation and appropriate normalization and passed through an additional attention layer which decides what aspects of a "post" are worthy of attention in determining the risk. While this can be done by looking at the attention coefficients/weights assigned to each sub-vector/feature, one can also manually add/remove features to assess the importance of features by looking at the empirical improvement/deprecation in performance.

As alluded to earlier, this is a post-level implementation and hence corresponds to "noisy copies" of a user. Since we have unique labels only at the user level, a post-level implementation unrolls the posts and copies the label for each post that belongs to a particular user. We expect this to deteriorate the performance and hope to implement a user-level classification in future work.

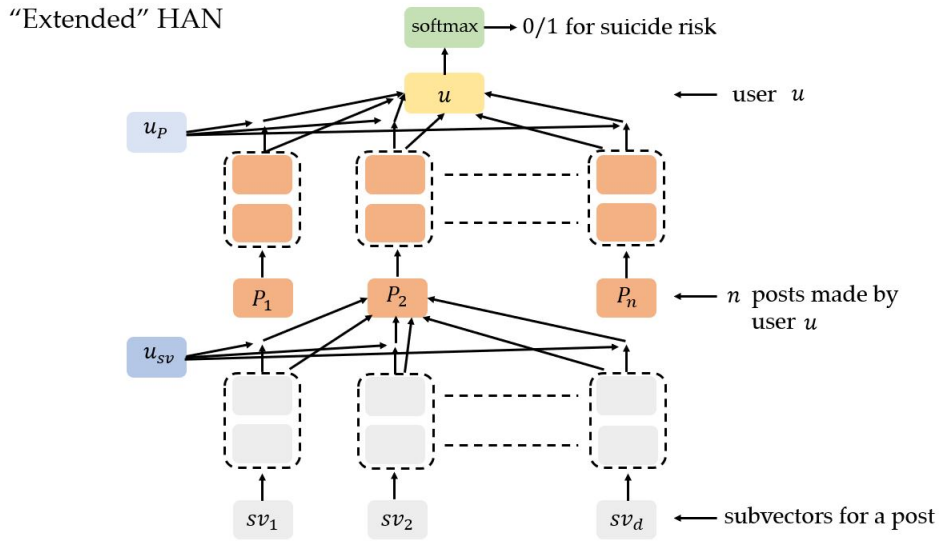

Figure 4: Extended HAN for user-level classification

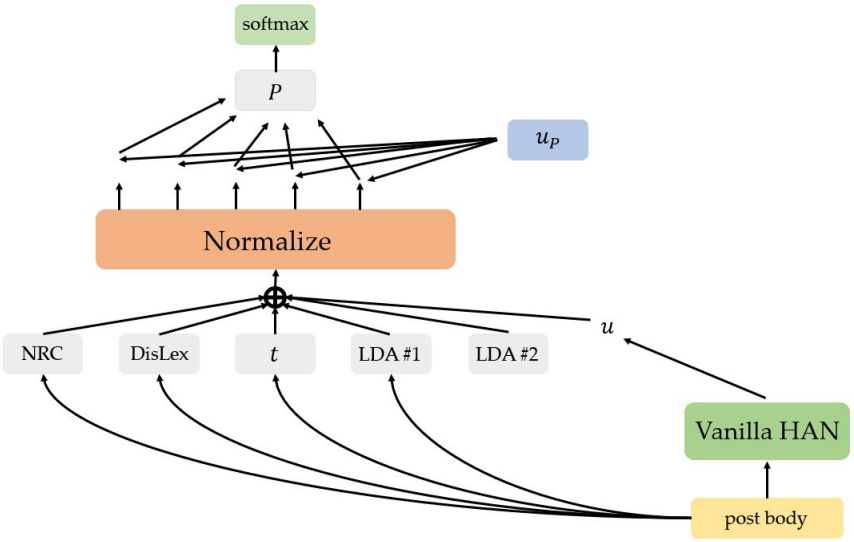

Figure 5: Current implementation (post-level)

[5]

5 Results

We trained our extended HAN model for post-level classification using Emotion, Mental disease Lexicon, user-level LDA on subreddit names, post-level LDA on post title and post body and the timestamp as features. The HAN model implementation is based on the one in [2]. We used the same cross entropy loss function and the ADAM optimizer to train our extended HAN model. But to avoid

[5] GitHub repository for implementation, https://github.com/angmavrogiannis/CMSC723-Project

over-fitting we simplified the model by reducing the number of Bidirectional GRU nodes and the fully connected layer nodes. The hyper parameter tuning for both batch size and the learning rate was done by using a grid search across [32,64] for batch size and [1e-3, 1e-4 and 3e-4] for the learning rate. The best classification metrics were obtained for the 64 batch size and the 1e-3 learning rate. We trained all the models for 20 epochs with "Early stopping" criteria based on the validation loss and with a patience of 3 epochs.

5.1 Classification metrics with individual features

Feature	Accuracy	AUC-ROC	Confusion matrix	Precision (High/Low)	Recall(High/Low)	F1 score(High/Low)
Emotion	0.53	0.50	[[0.68 0.32] [0.68 0.32]]	0.50/0.50	0.32/0.68	0.39/0.58
Mental Disease Lexicon	0.55	0.50	[[0.81 0.19][0.80 0.19]]	0.50/0.50	0.19/0.81	0.28/0.62
User-level LDA	0.53	0.50	[[0.69 0.31][0.69 0.31]]	0.50/0.50	0.31/0.69	0.39/0.58
Post-level LDA	0.55	0.51	[[0.75 0.25][0.72 0.28]]	0.52/0.51	0.28/0.75	0.37/0.61
Timestamps	0.56	0.50	[[0.90 0.09][0.90 0.09]]	0.51/0.50	0.10/0.91	0.16/0.65

Table 5: Classification metrics for individual features. Here "High" refers to the high risk class and "Low" refers to the low risk class

5.2 Classification metrics with all the features

Class	Precision	Recall	F1 score	Accuracy	AUC-ROC	Confusion matrix
High risk	0.56	0.29	0.38	0.57	0.53	[[0.78 0.22] [0.71 0.29]]
Low risk	0.52	0.78	0.63			

Table 6: Classification metrics from extended HAN model with all the features

6 Discussion: shortcomings, outlook, and analysis

Results in Table 5 and Table 6 shows that the extended HAN model we trained does not give promising results as expected. We speculate several reasons for the diminished performance in the neural language model. One of the key contributing factors is the implementation of a post level classification instead of a user-level classification. We think the post level classification is really challenging due to the fact that it contains a lot of noise which deteriorate the overall model performance. One other reason can be the difference in training and test set distributions where the train and test sets correspond to data collected during different years (from 2008 to 2015).

We believe some other technical details pertaining to the architecture details may also have played a part. For instance, the text (post_body) embeddings required the text to have a certain number of sentences for each post (otherwise would be padded), and hence we suspect filtering out smaller posts (with fewer than a cutoff number of sentences) should have boosted the performance.

With regards to the understanding the importance of each feature sub-vector (text encoding, LDA, timestamp, etc.) we have resorted to evaluating performance after adding/dropping features manually as opposed to examining the post level attention coefficients. Due to poor performance of the model, we noticed the attention coefficients fail to reveal any underlying systematics. We strongly believe that there is significant scope for improvement here and that attention coefficients should reveal important information in principle.

6.1 Error Analysis

To interpret the results, we thoroughly examine each feature we incorporated into the neural model and observe whether it is behaving as we expected.

We highlight a couple of important drawbacks of the NRC emotion lexicon features that we identified during error analysis. First, as mentioned before, only considering unigram emotion-related tokens does not take into consideration important bigrams (e.g. "happy" vs "not happy"). Second, a large portion of posts happen to concern gaming, and, given the nature of the games played (MMORPG,

FPS), the language for describing can be graphic (e.g. "kill", "destroy", "annihilate") or even swear words and phrases like "kill myself", but these words are actually referring to the in-game characters, rather than the users themselves, and hence confuse the pre-defined role of the NRC features. A clear example can be found at post_id='21wmth', which demonstrates very high anger and fear scores, when in reality the user is just referring to in-game content of the popular game League of Legends.

Another issue we observe can be attributed to the user-level LDA, as we identify cases where low-risk users happened to exhibit similar "posting" patterns with high-risk users in the subreddit level. Therefore, although we gain useful semantic information about their users through the subreddit names they posted at, we should probably shift our focus towards the LDA on the post body, which can also be evaluated intuitively by looking at the top words for each derived topic. On the other hand, the user-level LDA does not produce clearly segmentable subreddit-level topics.

Further, examining the misclassified posts, we note that an overwhelming majority of errors are committed for posts with label "d" or high risk. This implies that the classifier prefers to tag the post as being from a low risk user most of the time. We suspect this behavior is specific to the post-level implementation and can be alleviated in a user level one. Reason for this might be a large fraction of posts made in other sub-reddits (not r/SuicideWatch) by low-risk users which translate into multiple copies of low-risk posts.

7 Individual Contributions

- Anchit: Data Preprocessing, Post-level LDA
- Angelos: NRC emotion features, mentalDisLex, user-level LDA, interpretation of results
- Divya: Data preprocessing, LR baseline, Importance weighting, stopwords, bigram models
- Sanket: Data Pre-processing, Feature Engineering, Extended HAN Implementation and Tuning
- Yashish: SVM baseline, GloVe embeddings, Extended HAN implementation and fine-tuning, Interpretation of results

8 Data Use Disclosure

1. We have read Benton et al. (2017).
2. We understand that privacy of the users and their data is critical, and absolutely no attempt can be made to de-anonymize or interact in any way with users.
3. We understand that this project is being done solely for educational purposes, and the results cannot be used directly in research papers. If we get promising results and would like to develop the ideas into a research paper for publication, or to use what we have done further for another class, we will talk with Prof. Resnik about obtaining suitable Institutional Review Board review.
4. We understand that we may not use these data for any purpose other than this specific class project. We will not show or share this data with anyone outside class, nor do any research or development on this dataset outside the scope of the class project. If there are things we are interested in doing with this dataset outside the scope of the class project, we will talk with Prof. Resnik.
5. We will store the dataset and any derivatives on computers that require password access. If we are working in an environment where other people can log in, e.g. a department server, we will set file permissions restrictively so that only you have access.
6. Any copies of the data or derivatives of it will be accompanied by a clear README.txt file identifying Prof. Resnik as the contact person and stating further re-distribution is not to take place without contacting him first. If anyone we know is interested in the dataset, we will refer them to Prof. Resnik, rather than providing the data ourselves.
7. Once we have completed the project, we will delete any copy of the dataset we have made, including any derived files (e.g. tokenized versions of the documents).

8. We will not cut/paste any text content from this dataset into our project proposal, project writeup, onto the class discussion board, into e-mail, etc. If we want to identify a specific posting, e.g. in discussion on the class discussion board, we will use the ID from the dataset. If we want to give examples, we will create a paraphrase instead of the original text.
9. We have deleted all copies of the project dataset

References

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[2] Hee-Eun Lee Maria Kränkel. Text Classification with Hierarchical Attention Networks, 2019.

[3] Saif M. Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon, 2013.

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[5] Michael Röder, A. Both, and A. Hinneburg. Exploring the space of topic coherence measures. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 2015.

[6] Han-Chin Shing, Suraj Nair, Ayah Zirikly, Meir Friedenberg, Hal Daumé III, and Philip Resnik. Expert, crowdsourced, and machine assessment of suicide risk via online postings. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*, pages 25–36, New Orleans, LA, June 2018. Association for Computational Linguistics.

[7] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California, June 2016. Association for Computational Linguistics.

[8] Ayah Zirikly, Varun Kumar, and Philip Resnik. The GW/UMD CLPsych 2016 shared task system. In *Proceedings of the Third Workshop on Computational Linguistics and Clinical Psychology*, pages 166–170, San Diego, CA, USA, June 2016. Association for Computational Linguistics.