CS230

Detecting Flooding in Social Media Imagery Using Multimodal Deep Learning

Tamika J. Bassman
Civil and
Environmental Engineering
tbassman@stanford.edu

Usman Hanif
Computer Science
and Economics
uhanif@stanford.edu

Evelyn Xia
Statistics
evexia@stanford.edu

Abstract

Novel automated methods of detecting evidence of natural disasters such as floods from social media imagery have the potential to help first responders more rapidly understand severity of damage and identify impacted areas. In this work, we explore multimodal deep learning approaches to flood identification in social media images by means of image-metadata fusion. Several methods for fusion of metadata with a baseline, image-only CNN are presented and their results compared. Findings include that CNN- or LSTM-based metadata models do not rival the well-performing baseline image model given the modest size of the available metadata-set, and hence add little performance benefit through fusion of their predictions with those of the baseline image model. In contrast, fusion of deliberately chosen binary features extracted from the metadata is found to be capable of small accuracy improvements over the baseline image model.

1 Introduction

Floods are one of the deadliest and costliest forms of natural disasters, and pose a threat to a growing number of areas due to climate change and sea level rise-related effects [12]. People rely on various news and information outlets to communicate messages and gain knowledge of damage severity and impacted areas. Social media is increasingly utilized for rapid harvesting of information in several disasters over the last decade, including in New Orleans, Louisiana, during Hurricane Isaac in 2012, and in Victoria, Australia, during the 2010-2011 flooding events [1]. The sheer volume of publicly available social media data and the time-critical conditions of emergency response together spur the need for novel automated methods of identifying data which pertain directly to the disaster event of interest.

In this work, we propose multimodal deep learning approaches to identifying evidence of flooding events in image data from social media. Concretely, using images, their associated metadata, or a fusion of both, the target output of the algorithm is a correct label of either "flooding" or "no flooding" for each example. Our work builds off the framework of the 2017 MediaEval Multimedia Satellite Task 1, "Disaster Image Retrieval from Social Media" [8]. We begin with a baseline CNN model for flood detection that takes in only the images and utilizes a modified state-of-the-art image classification architecture, ResNet-18. We then attempt to improve this baseline performance by involving the metadata alongside the image data as model inputs, using a suite of different model architectures and fusion approaches.

2 Related Work

Entries to the original 2017 MediaEval competition cover a variety of approaches to this task. In terms of the image model, several authors [4, 6, 5, 7] extracted and refined features from multiple pre-trained and well-known deep CNN architectures and used support vector machines and late-stage fusion for image classification. Other researchers [14, 15] opted to replace the last layer of a single architecture such as GoogleNet or InceptionV3 with a two-neuron layer corresponding with a "flooding"/"non-flooding" binary output prediction; we draw upon this latter approach for our image-only baseline model.

CS230: Deep Learning, Spring 2021, Stanford University, CA. (LaTeX template borrowed from NIPS 2017.)

In terms of metadata fusion, some researchers [4, 5] utilized the Random Forest classifier from the WEKA library to output posterior probabilities for metadata classification, and these results were fused with the separate image predictions in late-stage post-processing. Random Forest generally exhibited poorer performance than approaches such as that of Lopez-Fuentes et al. [14], in which metadata were transformed into GloVe-initialized word embeddings and passed through a bidirectional LSTM, whose text features were concatenated with image features and fused via passage through subsequent fully connected and softmax layers. Model 3b in our work draws from this approach. Others [7, 15] followed a similar concatenation-based fusion approach, but trained Word2Vec-based embeddings on only one out of the sixteen total metadata fields and created a a word dictionary from two metadata fields, respectively.

In our work, we deviate slightly from the specific objectives outlined for the 2017 MediaEval competition and choose to focus in particular on exploring improvements to a baseline image-only model through fusion of the metadata, rather than through exhaustive hyperparameter tuning of the image-only workflow. Our contributions to this area of study include binary feature-based methods for metadata fusion that appear to be novel approaches to this particular task based on our literature review.

3 Dataset and Features

The social media dataset used in this study is curated for the 2017 MediaEval competition [8]. It consists of 6,600 (5,280 train/1,320 test) RGB Flickr images each pre-labeled with a "flooding"/"no flooding" label and accompanied by a set of metadata about the picture's contents, time and location of creation, and user-creator. Of the 6,600 total examples, 4,200 are labeled "no flooding" and 2,400 are labeled "flooding". We shuffled and subdivided the original 5,280-example training set into an 80/20 split (4,224/1,056) to serve as training and development sets, respectively.

3.1 Image Data

Images in the original dataset vary in dimensions, with the majority having a resolution of approximately 500x375x3 or 375x500x3, depending on whether the image was captured in portrait or landscape configuration. We transformed all images to have square spatial dimensions of 256x256 for uniformity. Some examples of transformed images in the dataset and their respective ground-truth labels are shown in Figure 1. To help counteract model overfitting, we also applied random horizontal flipping on the training set images.

Figure 1: Examples of images in the dataset with accompanying ground-truth flood/no flood labels.

3.2 Metadata

An example of the metadata fields accompanying a single image is shown in Figure 4. Our initial pre-processing of the image metadata involved removal of all metadata fields that we deemed irrelevant to the task, which resulted in retaining only three fields: "title", "description", and "user_tags". We also converted all "None" values to empty strings. Then, in order to prepare for vectorization, we cleaned the metadata by de-capitalizing and removing any remaining irrelevant parts such as punctuation, special characters, hyperlinks, image paths, image names, and stop words.

In two of the models constructed in this work (Models 2a and 2b, discussed in Section 4.2), three binary features were computed for each metadata example. Spricifically, they corresponded with whether or not each of the three fields "Description", "Title", and "User_tags" contained at least one of the following flood-related keywords: *flood*, *floods*, *flooded*, or *flooding*. We chose these keywords through manual inspection of the dataset with the intent of flagging most explicit indications of flooding without generating excessive false positives (e.g., from inclusion of contextually related but less targeted keywords such as *rain* or *damage*).

In the remaining two models constructed in this work (Models 3a and 3b, discussed in Section 4.3), these same three pre-processed metadata fields were encoded using pre-trained GloVe word embeddings [19], specifically the 50-dimensional versions of these vectors.

4 Methods

4.1 Model 1: Baseline Image Model

The architecture of our baseline image-only flood detection model is a CNN based on the ResNet-18 architecture [13]. It uses transfer learning to initialize the network with ResNet-18 pretrained weights for training. The final fully connected layer of the original network is modified to have an output dimension of 2, and a subsequent softmax layer provides the predicted probability of flooding for each image sample, whose error we measure using a cross-entropy loss function:

$$L = -\frac{1}{m}\sum_{i=1}^{m} y_i log(\hat{y}_i)$$

Our choice of this approach stems in part from our observations across the 2017 MediaEval submissions, in which single-architecture models were able to outperform ensemble learning approaches. A ResNet-18-based architecture also had the highest "Average Precision at 480" metric in the image-only sub-task of the competition [15].

4.2 Models 2a, 2b: Metadata Fusion via Binary Features

The first two model architectures for multimodal data fusion make use of the binary features of the metadata extracted during pre-processing as described in Section 3.2. The motivation for developing these architectures was to explore whether fusion of relatively simple, manually constructed features determined from inspection of the dataset is capable of improving the baseline model performance.

To create the architecture of Model 2a, several layers are added between the last fully connected layer and the ultimate softmax layer of Model 1, as shown in Figure 2. Of note, the image feature vector and 3x1 binary feature vector are separately batch-normalized prior to concatenation to ensure comparable orders of magnitude of values across the two vectors. We introduce a scalar hyperparameter designated α_2 to enable tuning of the weight of the binary features relative to the weight of the image features. Immediately prior to concatenation, the binary feature vector is scaled elementwise by a factor of $1 + \alpha_2$ and the image feature vector by a factor of $1 - \alpha_2$.

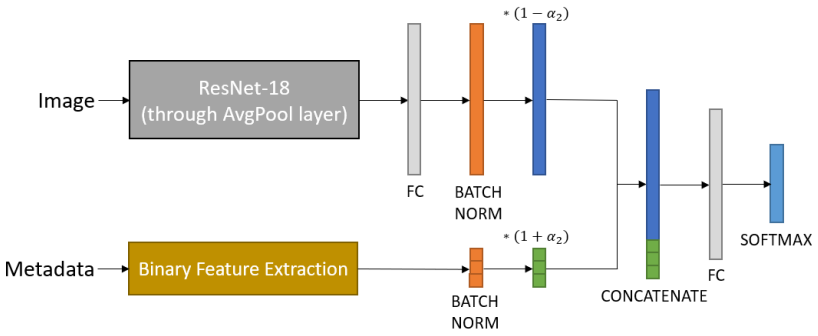

Figure 2: Schematic of Model 2a architecture.

To create Model 2b, a single modification is applied to the flooding/no flooding prediction outputted by the softmax layer of Model 1. Let *p* represent the softmax layer's outputted probability of flooding (prior to modification) and let the subscripts *ti*, *d*, and *ta* represent the title, description, and user tag fields of the metadata, respectively. The final probability of flooding outputted by Model 2b has the value

$$\min(\max(\alpha_{comb} p, 0), 1) \tag{1}$$

where

$$\alpha_{comb} = \prod_{i \in \{ti,d,ta\}} (1 + \alpha_i \mathbf{1}_i - \beta_i(1 - \mathbf{1}_i)) \tag{2}$$

and

$$\mathbf{1}_i = \begin{cases} 1, & \text{metadata field } i \text{ contains flood keyword(s)} \\ 0, & \text{metadata field } i \text{ does not contain flood keyword(s)} \end{cases} \tag{3}$$

Hence, the value of α_{comb} is dependent entirely on the choice of hyperparameters and the metadata binary feature values. For positive values of the six hyperparameters α_i and β_i, each of the three α_i hyperparameters act to increase the

confidence in flooding only if their respective field contains a flooding keyword, and each of the three β_i hyperparameters act to decrease the confidence only if their respective field does not have any flooding keyword.

4.3 Models 3a, 3b: Metadata Fusion via Word Embeddings and Additional Neural Networks

In an attempt to further increase accuracy on our metadata predictions, we implemented a CNN and an LSTM-based RNN model for metadata classification. Applying CNN to word embeddings has proven to be successful in finding general patterns and performing text classification. On the other hand, the driving force behind choosing an RNN method was its ability to store internal state memory, which is not possible with the binary features model expressed above.

The architecture of the CNN component of Model 3a consists of an embedding matrix layer, a 1D CNN comprising multiple blocks of convolutional layers and maxpooling layers, multiply fully connected layers, and a dropout layer. The architecture of the LSTM component of Model 3b consists of an embedding matrix layer and an LSTM model with one embedding layer, one LSTM layer, and one fully connected layer. For both Models 3a and 3b, the CNN and LSTM are trained separately of the image baseline, and the predicted labels for each example outputted by the 1D CNN and the LSTM, respectively, are late-stage fused with the baseline image model via concatenation at an analogous location in the architecture to the one shown in Figure 2 for Model 2a.

5 Experiments and Results

Accuracy and average precision metrics were computed to evaluate the performance of each of the models and are summarized in Table 1. Average precision is relevant to social media image detection by considering both the accuracy of the final outputted binary label as well as the confidence of the model in each flooding prediction, which serves as a proxy to the order or ranking of flooding images returned by the algorithm when hypothetically deployed [8]. Confusion matrices for all five models are presented in Figure 5, and precision-recall curves in Figure 6.

Table 1: Summary of converged results for all 5 models. (Acc = accuracy, Avg Prec = average precision)

Model	Train Acc	Train Avg Prec	Dev Acc	Dev Avg Prec	Test Acc	Test Avg Prec
1	1.000	1.000	0.931	0.958	0.925	0.953
2a	0.984	1.000	0.937	**0.970**	0.933	**0.962**
2b	1.000	1.000	**0.943**	0.960	**0.938**	0.938
3a	0.984	1.000	0.933	0.963	0.920	0.938
3b	0.984	1.000	0.934	0.963	0.921	0.941

5.1 Model 1

For the purpose of the baseline, we employed choices of hyperparameters suggested as a starting point for image classification transfer learning applications in the literature [9]. We used mini-batch gradient descent optimization with a batch size of 32, an initial learning rate of 0.001, learning rate decay with a step size of 7 and a decay parameter of 0.1, and a momentum factor of 0.9. The performance of our baseline model was found to be relatively high with minimal hyperparameter tuning, and as our primary objective in this work is to investigate performance improvements by means of metadata fusion, subsequent effort was spent developing the remaining four metadata-based modeling approaches described below.

An occlusion sensitivity study per [21] of a sample of test set images which the baseline model correctly predicted, as shown in Figure 3, revealed that the image model appears to detect flooding by picking up on edges or objects which interrupt large, relatively uniform segments of water geometrically or in texture. The absence or erroneous detection of such edges in other manually examined images was found to lead to false negatives or false positives, respectively, in some cases.

5.2 Models 2a, 2b

Experiments for tuning of hyperparameters α_2, α_i, and β_i are summarized in Tables 2 and 3. Models 2b and 2a were found to have roughly 1% improvements to the baseline performance in terms of accuracy and average precision, respectively, on the test set. From the confusion matrices shown in Figure 5, it is apparent that a reduction in both type 1 and type 2 errors relative to the baseline contributes to this improvement in the case of Model 2b. Manual error analysis revealed that further improvement to the baseline may be hindered by the current design of the binary features. For

Figure 3: Occlusion sensitivity analysis of five test set examples labeled "flooding". All 19x19 occlusions (separated by a stride of 6) which individually caused the baseline model to erroneously predict “no flooding” are superimposed in the bottom row images.

example, phrases such as "not flooding" appear in the metadata for some negative flooding examples, and some positive examples do not contain any of our four chosen keywords listed in Section 3.2.

5.3 Models 3a, 3b

In the process of developing Model 3a, we constructed a fully connected neural network that has one hidden layer, and whose input was a concatenated vector of the average word embedding for each of the three metadata fields of interest. This fully connected model yielded 0.5567 accuracy on the testing data, as it largely suffered from overfitting. We then pivoted to preparing the 1D CNN model alternative designated Model 3a. For hyperparameter tuning, we adjusted the dropout rate, dimensions of hidden layers, and the batch size. This model yielded 0.5992 accuracy on its standalone metadata classification (prior to fusion with the image model), which was slightly better than the fully connected model but still far below the image baseline. Hyperparameter tuning of the LSTM in Model 3b involved varying the number of fully connected layers and the dropout rate, and introducing a bidirectional LSTM. The best standalone metadata classification accuracy achieved with the LSTM was 0.7528.

Following fusion of the CNN and LSTM predictions with the image pipeline of Model 1, Models 3a and 3b offered no performance gain over the image baseline, as evident from Table 1 and the slightly steeper downsloped precision-recall curves in Figure 6. The only slightly worse performance relative to Model 1 suggests the effect of the metadata contributions fused during concatenation is progressively reduced or ignored during training of Models 3a and 3b.

6 Conclusions and Future Work

In this work, we have demonstrated the potential for fusion of metadata to improve the accuracy of deep neural network-based image classification algorithms for flood detection in social media imagery. Within the context of NLP problems, the dataset available for this work was on the smaller side (i.e., on the order of a few thousand examples). In such a case of limited training data, our results demonstrate that manually determining relatively simple but targeted binary features from the metadata was more effective at improving the accuracy of the multimodal model than a relatively more complex approach in which the model attempted to learn the most important features of the metadata itself, such as through training a word embedding-based CNN or LSTM model. With a larger set of training data and more extensive hyperparameter tuning, this latter, more end-to-end approach to fusing image data and metadata might yield better results.

Further development of this work would involve gathering more data for the training set particularly to improve the overfitting of all models evidence in Table 1. We also would refine the pre-processing of the metadata and generation of word embeddings for Models 3a and 3b. We would perform more extensive hyperparameter tuning of all architectures explored herein, including improvements aimed at reducing overfitting to the training set. For the binary feature-based components of Models 2a and 2b, we would explore the available metadata further to identify whether there are any additional flooding keywords that should be considered in the determination of binary features. Lastly, we would consider trying alternative fusion approaches entirely, such as early-stage fusion or joint/simultaneous training of the image- and metadata-based neural networks.

Contributions

All team members performed literature review; wrote portions of the proposal, milestone, and final report; and co-produced the video. Usman pre-processed the metadata and coded and tuned the LSTM for Model 3b. Evelyn pre-processed the metadata and coded and tuned the CNN for Model 3a and the additional fully connected model that was also tried for training on solely the metadata. Tamika coded and tuned Models 1 and 2a/b, incorporated the Model 1 workflow into Models 3a/b, and prepared the results.

Acknowledgements

The authors thank Shubhang Desai and Huizi Mao for their advice in the initial stages of this project and throughout this project's development, respectively. The baseline image model, binary feature-modified models, and LSTM model were implemented using PyTorch [17]. The fully connected and CNN-based metadata models were implemented using Keras [11] and TensorFlow [3]. Precision-recall results were computed and visualized using scikit-learn [18] and the Github repo developed by Dennis Trimarchi [20][1]. The authors further thank previous CS 230 teaching staff for preparation of the course Github repo [2], the PyTorch tutorial prepared by Chilamkurthy [9], the LSTM tutorial prepared by Aakanksha [2], the LSTM tutorial prepared by Pai [16], and the Keras tutorial prepared by Chollet [10].

References

[1] Innovative uses of social media in emergency management. https://www.dhs.gov/sites/default/files/publications/Social-Media-EM_0913-508_0.pdf, 2013.

[2] NS Aakanksha. Multiclass text classification using lstm in pytorch. https://towardsdatascience.com/multiclass-text-classification-using-lstm-in-pytorch-eac56baed8df, 2020.

[3] M. Abadi et al. Tensorflow: Large-scale machien learning on heterogeneous systems. https://www.tensorflow.org/, 2015.

[4] K. Ahmad et al. Cnn and gan based satellite and social media data fusion for disaster detection. *MediaEval'17*, 2017.

[5] S. Ahmad et al. Convolutional neural networks for disaster images retrieval. *MediaEval'17*, 2017.

[6] K. Avgerinakis et al. Visual and textual analysis of social media and satellite images for flood detection @ multimedia satellite task mediaeval 2017. *MediaEval'17*, 2017.

[7] B. Bischke et al. Detection of flooding events in social multimedia and satellite imagery using deep neural networks. *MediaEval'17*, 2017.

[8] B. Bischke et al. The multimedia satellite task at mediaeval 2017. *MediaEval'17*, 2017.

[9] Sasank Chilamkurthy. Transfer learning for computer vision tutorial. https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html.

[10] F. Chollet. Using pre-trained word embeddings. https://keras.io/examples/nlp/pretrained_word_embeddings/, 2020.

[11] F. Chollet et al. Keras. https://keras.io, 2015.

[12] T. Frank. Floods are increasing in supposedly low-risk areas. https://www.scientificamerican.com/article/floods-are-increasing-in-supposedly-low-risk-areas/, 2021.

[13] K. He et al. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[14] L. Lopez-Fuentes et al. Multi-modal deep learning approach for flood detection. *MediaEval'17*, 2017.

[15] K. Nogueira et al. Data-driven flood detection using neural networks. *MediaEval'17*, 2017.

[16] A Pai. Build your first text classification model using pytorch. https://www.analyticsvidhya.com/blog/2020/01/first-text-classification-in-pytorch/, 2020.

[1] https://github.com/DTrimarchi10/confusion_matrix

[2] https://github.com/cs230-stanford/cs230-code-examples

[17] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems 32*, 2019.

[18] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[19] J. Pennington et al. Glove: Global vectors for word representation. `http://nlp.stanford.edu/data/glove.6B.zip`, 2014.

[20] D. Trimarchi. Confusion matrix visualization. `https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea`, 2019.

[21] M. Zeiler et al. Visualizing and understanding convolutional networks. *ECCV 2014: Computer Vision*, 201.

Appendix


```
{ "images":[{ "image_id": "12328463323",
              "image_url": "http://www.flickr.com/photos/9752474@N07/12328463323/",
              "image_extension_original": "jpg",
              "date_taken": "2014-01-30 10:18:12.0",
              "date_uploaded": "1391631137",
              "user_nsid": "9752474@N07",
              "user_nickname": "SurferJoe88",
              "title": "Emma Wood State Beach - campsites",
              "description": "Emma Wood State Beach - flooded campsites",
              "user_tags": ["flooding"],
              "license_name": "Attribution-NonCommercial-ShareAlike License",
              "license_url": "http://creativecommons.org/licenses/by-nc-sa/2.0/",
              "capture_device": "SAMSUNG PL70 / VLUU PL70 / SAMSUNG SL720",
              "latitude": 34.2871540000000102,
              "longitude": -119.32989299999999844},
              ...
            ]
}
```


Figure 4: Metadata fields accompanying each image in the dataset.

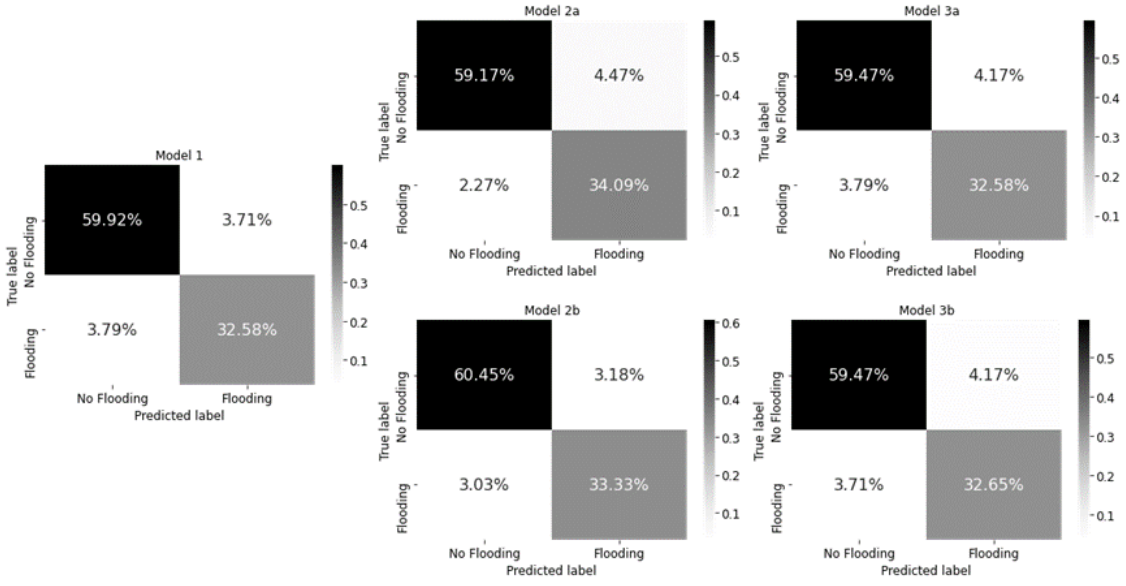

Figure 5: Confusion matrices for all 5 models.

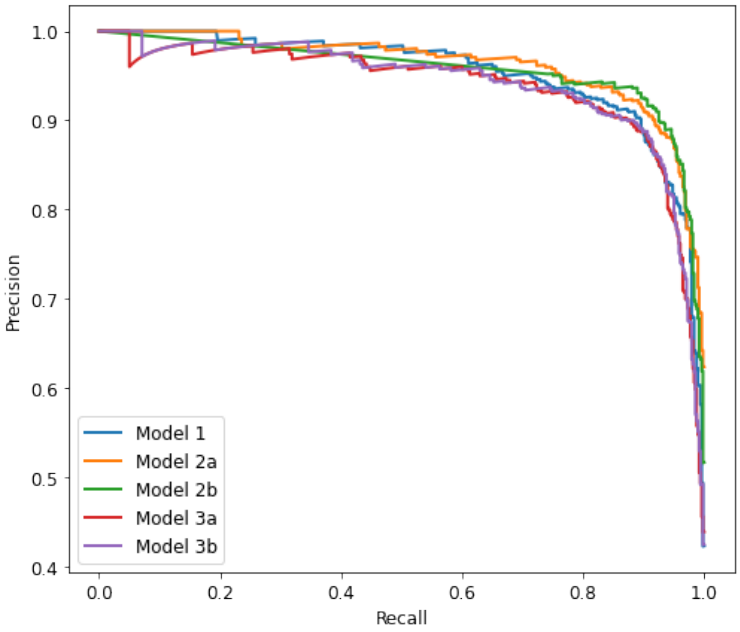

Figure 6: Precision-recall curves for all 5 models.

Table 2: Experiments on hyperparameter α_2 for Model 2a.

α_2	Dev Set Accuracy
0.8	0.8561
0.6	0.8826
0.4	0.9337
0.2	**0.9384**
0.01	0.9375
-0.1	0.9366
-0.5	0.9337

Table 3: Experiments on hyperparameters for Model 2b.

α_{ti}	α_d	α_{ta}	β_{ti}	β_d	β_{ta}	Dev Set Accuracy
0.5	0.5	0.5	0	0	0	0.9328
0	0	0	0.5	0.5	0.5	0.6714
0.5	0.5	0.5	0.05	0.05	0.05	0.9356
0.5	**0.5**	**0.5**	**0.05**	**0.05**	**0.5**	**0.9432**
0.5	0.5	0.5	0.05	0.05	0.6	0.9347
0.5	0.5	0.5	0.05	0.05	0.4	0.9422