TwitchMoji: Explainable Sentiment Classification using Self-Supervised Labels and Prototypes

Final Project for CS378: Natural Language Processing with Dr. Eunsol Choi

Yian Wong Randy Yu

Abstract

We introduce a completely self-supervised sentiment classification dataset consisting of chat messages from the popular streaming service, Twitch. In this dataset, we use emotes as a weak form of labels indicating sentiment and emotion. On this emote classification task, we introduce a white-box model that is heavily based on Prototype Tensor Explainable Networks (ProtoTEx). Prototype networks encodes and clusters input data and implicitly learns cluster heads that the encoded input data are centered around.

Prototype networks can then classify only using an encoded input text's distances from all of the model's prototypes, and by doing so add a critical layer of explanation through providing the user with the closest training examples in the encoding latent space. We show that by adding prototype layers, we achieve similar or better performance in this task compared to the pre-trained baseline BART classification model. We also show that we can easily map prediction scores on Twitch emotes directly to corresponding sentiment labels from the task we worked on in our NLP class and achieve resuts better than models we studied in class.

1 Introduction

NLP tasks are often faced with the practical problem of lack of annotated data. As a result, many attempts have been recently made to create tasks that are self-supervised, or altogether don't require any labeling. We present the Twitch Chat Emote dataset, collected through Twitch's API on recent stream and chat replays. This is similar to work done in Li et al where tweets from Twitter were collected. As opposed to Twitter, we believe Twitch chat gives users a higher level of interactivity between each other and the stream. Streams can show a vast range of emotions, and coupled with Twitch's unique emote dataset, we believe that Twitch users convey a great amount of emotion as streams go through different emotions and experiences. As a result, this project's aim is to gather more insight into the emotions shared by users on Twitch through the chat.

Another problem we aim to tackle in this project is enabling models to explain their decisions. Often times in many NLP tasks (like hate speech detection or rhetoric classification), understanding model decisions require expert domain knowledge, which makes it difficult for users to understand and verify models. In the Twitch Chat Emote dataset, we classify over an extremely diverse set of emotes, each of which can be used in several specific situations, and possibly characterizing very different emotions and sentiment for users. To address this issue of common neural network based NLP models not being able to explain their decisions, we introduce Prototype networks, based on the ProtoTEx model in Das et al.

We train a classifier that uses BART encodings as baseline models on this dataset. We then introduce our prototype-based network and evaluate it on the same task, similarly to BART. We present empirical results showcasing its ability to explain and provide similar examples, while maintaining comparable classification performance to baseline models. We also show that by learning these emotes, we can use the trained models to map to labels in other sentiment classification tasks with high performance.

2 Related Work

The task used in this work is heavily based on work done in (Felbo et al, 2017), where they predict emojis used in Tweets. Felbo et al show that they can use the top 64 regular emojis in their dataset as labels for Tweets, and train an LSTM-based model with custom vocabulary to predict and extent their models to predict other classes of sentiment, emotion, and sarcasm across different domains and datasets. We intent to extend this method to Twitch emotes and show they have as much, if not more diversity in terms of meaning and sentiment as emojis. Felbo et al is the most notable recent successful use of emoji prediction for robust sentiment analysis across multiple domains.

We also leverage the use of prototype networks, introduced in (Li et al 2017), coupled with the BART model (Lewis et al 2019). The coupling of these two architectures was first introduced in (Das et al 2022), a model called ProtoTEx. This model was traditionally used in propaganda and hate speech detection, but we research a similar model in emote prediction and sentiment analysis. Rather than using BART's encoder and decoder as part of the prototype, we treat the BART encoding as a non-trainable input to our prototype network. We thus create another autoencoder within this model to be able to freeze the BART encodings and model prototypes in smaller dimensional latent spaces. We couple research done in these works to explore the Twitch dataset and the explainability of the prototype network.

3 Task / Datasets

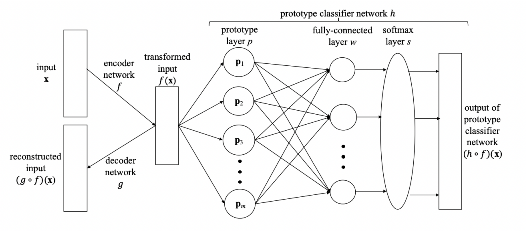
We curate an emote classification from Twitch. Twitch is a popular streaming service where viewers can interact with video streams via a chat. Twitch's chatting service is specifically unique because it as a whole receives over 1,200 messages a second at all times, and features a widely used emoticon set that is widely used across all genres and channels on the platform.

In this paper, we use Twitch's API to gather chat messages with emotes used in them, giving us about 400 million chat messages. Our task is to predict the emotes used in a given text chat. In every chat message, we remove the emote and make it it's label.

Figure 1: Examples of emotes frequently used in Twitch. Emotes are denoted in the chat by typing a special case-sensitive keyword (ex: 'Kreygasm', 'BibleThump')

To keep this problem as a single label classification task, we assign only one chat message a single label. If a chat text has multiple emotes, we add multiple instances of that text, where each added instance has a different label for the multiple emotes.

To ensure our models do not learn a bias from the distribution of emotes in the dataset, we re-sample the dataset such that every emote class has an equal chance of being picked. Additionally, while Twitch features over a thousand diverse emotes, we only use the most common 64 emotes in our research in this paper.



prototype classifier network h
prototype layer p
fully-connected layer w
softmax layer s
input $\mathbf{x}$
encoder network f
transformed input $f(\mathbf{x})$
reconstructed input $(g \circ f)(\mathbf{x})$
decoder network g
$\mathbf{p}_1$
$\mathbf{p}_2$
$\mathbf{p}_3$
$\mathbf{p}_m$
output of prototype classifier network $(h \circ f)(\mathbf{x})$

Figure 1: A generic prototype network. In our work, we use this exact architecture, and use BART encodings as representations for our input texts. These encodings therefore are the input to our prototype network, x

4 Model

We use BART as our baseline model, using its encoding of the text as inputs to a logistic classifier.

For our prototype network, we use BART encodings as representations for an input text, o. The original prototype paper uses an autoencoder to encode input into a compact latent space, and models several prototype tensors, p_i, within this latent space which represent cluster heads of the data. We denote the input text BART representation as as x, its encoding as $z(x)$, and the reconstruction produced by our autoencoder as as $r(z)$. We train the prototype network to optimize the following three losses:

$L_{rc}(\theta) = \frac{1}{k}\sum_k \frac{1}{2}||x_k - r(z(x_k))||_2^2$

$L_1(p) = \frac{1}{m}\sum_{j=1}^{m} min_{i\epsilon[1,n]}||p_j - f(x_i)||_2^2$

$L_2(p) = \frac{1}{n}\sum_{i=1}^{n} min_{i\epsilon[1,n]}||p_j - f(x_i)||_2^2$

These losses have been shown to enable prototypes to encode rich features about the data in a completely unsupervised way, forcing the model to cluster data around at least one of the prototype tensors. Minimizing L_1 requires all prototypes to be closer to at least one of the training examples. Minimizing L_2 requires every training example to be closer to at least one prototype. Furthermore, L_{rc} is the autoencoder's reconstruction loss, which forces the encodings to preserve its original information.

In addition to these losses, we also optimize the cross entropy loss between the labels y and the prediction of the network's logistic classifier $\hat{y}$:

$L_{ce}(\theta) = \frac{1}{n}\sum_{i=1}^{C} -y_i log(\hat{y_i})$.

Hence, our final loss is:

$L(\theta) = \lambda L_{rc} + \lambda_1 L_1 + \lambda_2 L_2 + L_{ce}$

For all experiments, we freeze every layer of BART and treat it strictly as a non-trainable encoder for input text. We treat BART's last hidden state as the encoding, which has a hidden size of 256, and compress it into a latent state of size 64. In the prototype layer, we compute the two-norm distance of the vector from every prototype. These distances are used as inputs into our logistic classifier, which then predicts one of the top 64 emotes for its input.

5 Experiments

5.1 Experimental Setups

We now describe the details behind our experiment. For every training example, we tokenize it using BART's tokenizer, and use the tokens as inputs to the BART model. We take the last hidden state of the BART model and use them as inputs to the logistic classifier or the prototype network. We train using mini-batching with batch size of 1024 and use an early stopping policy on the validation score with a patience of 7 epochs. For both models, we use the Adam optimizer with a learning rate of 5e-5 with weight decay.

Additionally, we use our model as a model in the sentiment classification task in homework 1 of this class. We do this by freezing our entire prototype network and using the predicted logits of the 64 classes as an input to a logistic classifier. By fine-tuning this transfer-learned model, we effective can create a mapping between Twitch emotes and positive/negative sentiment.

5.2 Results

We find that in our analysis that the model performs very poorly quantitatively if we only look at its top prediction. This is because we collected our data such that one chat text with multiple emotes are considered different instances each with different labels. Hence, we find it more beneficial to consider the top k emotes, for some positive integer k. In for our analysis, we found it suitable to consider the top 5 predicted emotes in our analysis. In Table 1, we show the top 5 accuracy and top 5 F1 score of the two models.

Model	Top-5 Accuracy	Top-5 Macro F1-Score
BART	47.2%	0.45
Prototype BART	46.5%	0.44

Table 1: Accuracy and macro F1 score of model's predictions on a test subset of about 10 million examples

The difference in accuracy and F1 score between the baseline BART model and Prototype BART underlines ability for Prototype BART to enforce structure on the hidden parts of the neural network. We can effectively create a white-box structured neural network without significantly compromising classification accuracy and performance. Prototype models can give more meaning to the weights learned through gradient descent than otherwise learned in the baseline BART model.

6 Analysis

Analysis of accuracy and F1-score does not indicate the value of the dataset in helping models generalize to other domains or the explainability of the prototype network.

6.1 Generality of learning Twitch emotes

To quantify the value of emote prediction in other domains, we use a logistic classifier to map logits from the Prototype BART model's predictions to sentiment analysis from our first NLP homework. Below we show results on the sentiment analysis dataset as reported by the autograder, in comparison to a fine-tuned BART model not using logits as inputs, but the text as direct inputs:

Model	Accuracy	F1 Score
Logistic Regression	78%	0.79
Deep Averaging Network	77%	0.76
BART classifier	82%	0.84
Twitch pre-trained Prototype BART	87%	0.88

Table 2: Comparison of highest performing models with BART and Twitch pre-trained Prototype BART

As we can see, the learned emotes can be readily mapped using a logistic classifier to the sentiment classification task. We show through this small example that these 64 emotes show a diverse range of emotions and sentiment in texts, which allows models to transfer their knowledge to other domains and achieve top performance out of other conventional methods.

6.2 Explained model decisions using Prototype BART

Explanations are difficult to quantify, so we provide a qualitative analysis on empirically different sample texts and show the closest training examples so that we can empirically see that the model provides examples that are better explain its decision by showing examples with similar sentiment, emotion, and emote usage:

Model	Similar examples (from Prototype BART)
NA ULT	NA NADE
	BRAZIL MOLLY
	EU FLASH
Yooh FA?!	CAMELLIA TIEBREAK
	Wait Yooh is a featured artist?????
	Xi always makes the best songs
EL DIABLO	YAY IS THE GOAT
	THE BEST PLAYER IN THE WORLD
	EL DIABLO OMG
SUS	HUHH
	YOOOOO
	What did he say??
How that fsmash hit	THAT BAIR HIT
	These hitboxes man
	Wtf Nintendo...

Table 3: Sample chat messages and predicted training examples that are similar by Prototype BART model

A careful Twitch viewer will be able to identify that these are similar examples. To a naive reader, texts such as 'NA ULT' with an emote will not make much sense in terms of meaning, but upon seeing other examples like "NA NADE" and "EU FLASH", it becomes clearer that the chat message is likely making fun of a certain region (EU or NA) for a play that happened in the stream. Another example like "How that fsmash hit" might make no sense at first, but then comparing to the close examples, it seems that these chats are complaining or are shocked by how big the hitboxes are in a certain game made by Nintendo, probably Smash.

A user study on non-experts rating their understanding of chat with and without the Prototype BART model's similar examples will likely be a strong way to quantitatively assess the model's ability to explain its predictions. This type of study is left for future work.

7 Conclusion

We show that Prototype BART can effectively predict Twitch emotes in a large diverse dataset of chat texts. We show that adding prototype layers to BART allows the network to learn rich encodings and cluster heads for the data while maintaining comparable performance with regular black-box models. Although our model does not succinctly explain all of a model's decisions, we believe that adding this extra layer of context by providing similar training examples will enable it to provide non-experts with extra information in specific classification domains.

References

[1] Felbo, Bjarke et al. (2017) *Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm*

[2] Li, Oscar et al. (2017) *Deep Learning for Case-Based Reasoning through Prototypes:A Neural Network that Explains Its Predictions*

[3] Das, Anubrata et al. (2022) *ProtoTEx: Explaining Model Decisions with Prototype Tensors*

[4] Lewis, et al. (2022) *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*