CS230

Graphical Feelings: GIF Sentiment Learning

Gordon Blake
Department of Computer Science
Stanford University
gblake@stanford.edu

Abstract

With the growing popularity of animated GIFs on social media, the need for improved GIF search is evident. To further this goal, we experiment with a deep learning system that predicts real-valued sentiment scores for GIFs using two possible network architectures: a MaxPool fully connected model and an RNN using LSTM units. We train these models on the GIFGIF dataset, tune their hyperparameters, explore sampling methods, and report results. Both models tend to make conservative predictions close to the class means, suggesting that key sentiment features were not learned. CNN encoders more specialized to capture facial and mid-level sentiment features may be needed to improve performance. Code for this project is available on our GitHub repository (currently private).

1 Introduction

GIFs, brief animated images often of a storytelling nature, have become increasingly popular in recent years, used in communication through social media, email, and text messages [5]. However, users' ability to search for GIFs has been limited by their lack of categorization, which is often based on hand-tagging by humans. Users often seek GIFs that convey a certain sentiment, such as happiness or anger. Here, we explore an approach to automatically score the sentiment of a GIF using a CNN encoder and several possible decoder architectures. The input is a single GIF, consisting of a sequence of image frames. This is passed through a CNN encoder, then a decoder network, which outputs a vector of 17 real values between 0 and 1 representing sentiment scores on each of 17 emotional axes.

2 Related work

Perceived emotion prediction has been performed on the GIFGIF dataset by Jou et al. [4] using linear regression and feature representations chosen for their connection to emotion recognition. These features included the SentiBank [1] representation of mid-level visual features encoded as a 1200 dimensional vector and facial expressions captured by a small pretrained CNN. However, the authors did not experiment with the output of a more generalized CNN such as ResNeXt. The authors calculated GIF sentiment scores using the TrueSkill algorithm provided by the GIFGIF API, which was not available at time of writing. As discussed below, we instead scored GIFs using the Bradley-Terry model. Ji et al. [3] discuss the value mid-level sentiment ontologies such as SentiBank in GIF sentiment classification (not regression) on another GIF dataset known as the GSO Framework. These results suggest the value of using a CNN to encode high-level image features for learning. In general however, deep learning approaches to GIF sentiment scoring (particularly decoding the CNN output) have not yet been attempted. Xu et al. [6] found success using transfer learning on a trained CNN with several fully connected output layers in order to predict the sentiment of still images, indicating that such an architecture may generalize to sequences of images encoded by GIFs.

CS230: Deep Learning, Winter 2019, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

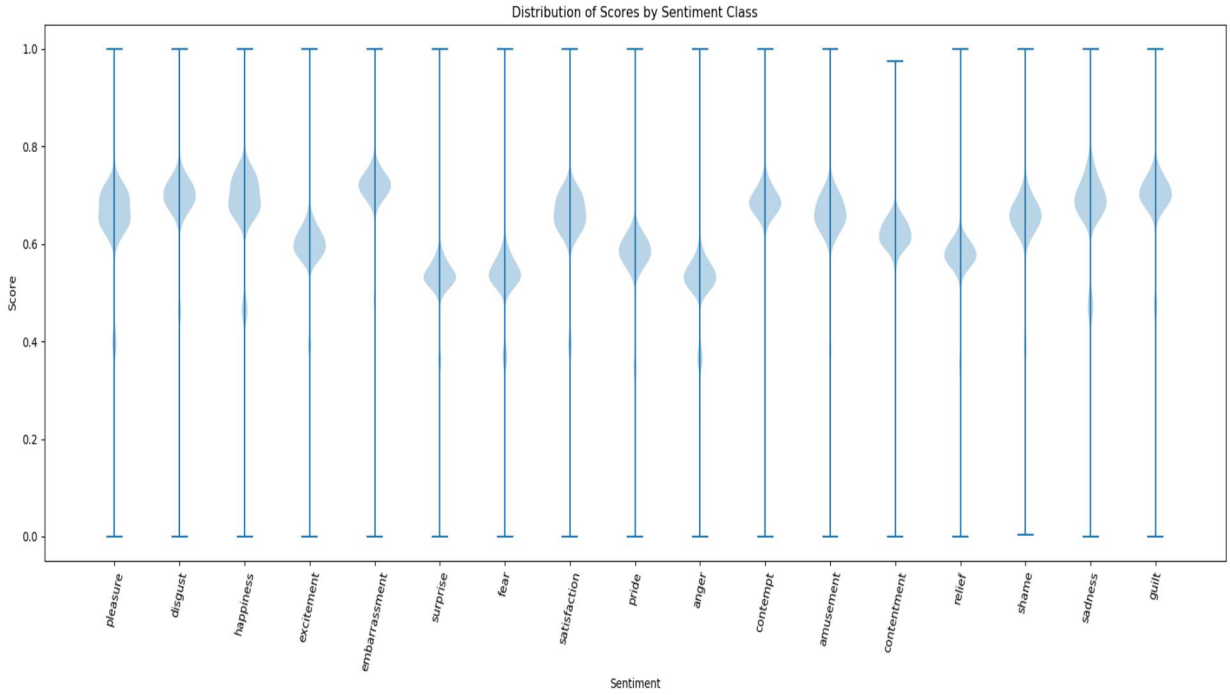

Figure 1: Distribution of GIF sentiment scores by emotion class.

Indeed, the CNN encoder/RNN decoder architecture has been used in the past for GIF captioning, notably by Li et al. [5], who applied both classical machine learning and deep learning to the problem. They found that encoding the frames of the GIF using the ImageNet CNN and feeding the resulting sequence to an LSTM produced relatively accurate GIF captions. Thus CNN encodings such as those created by ImageNet contain sufficient information to create rich GIF descriptions.

3 Dataset and Features

We used the GIFGIF dataset created by MIT researchers and available at GIFGIF Media. The dataset consists of 6143 GIFs from the Giphy API labeled with the results over 2.7 million pairwise comparisons given by internet users. For each comparison, users were presented two GIFs and asked which displayed more of a certain emotion, such as happiness. The users then could select either GIF or answer "neither." We used the choix implementation of the Bradley-Terry model to convert the pairwise comparisons into 17 real-valued sentiment scores for each GIF, which were then min-max scaled to range between 0 and 1. An example GIF scoring most highly on happiness is linked here. The distribution of the scores for each sentiment class is show below in Figure 1. Note that each score distribution is multimodal and consists of three normal distributions, with the majority of the data clustered around the uppermost distribution.

After obtaining the dataset, we divided it into an 80/10/10 train/dev/test split. GIFs with fewer than 4 frames were filtered from the dataset. In order to address potential variance problems, we also created an augmented training set by flipping, scaling, translating, and adding salt and pepper noise to the original training GIFs, resulting in an augmented training set with 43,995 images (but only 4,894 unique sentiment score vectors).

4 Methods

4.1 Encoding

We used the ResNeXt-101 CNN encoder pretrained on the Kinetics dataset, which produced the best results in experiments run by Hara, Kensho, Hirokatsu Kataoka, and Yutaka Satoh [2]. Implementation

details for the ResNext-101 CNN can be found in the source research paper. The CNN outputs features of 2048 dimensions. We modified the code to take GIFs as input instead of videos and experimented with several different sampling rates.

Initially, we produced encodings for every 1 in 10 frames of each GIF in order to align with the model in Li, Yuncheng, et al. [5]. However, finding that this was an insufficient sampling rate for most GIFs, as the median number of frames in a GIF was 20, we implemented two different sampling schemes. In the sparse sampling scheme, we ensured that all GIFs had at least three frames sampled by taking the first, last, and middle frames for GIFs with fewer than 30 frames. Longer GIFs were sampled at a rate of 1 in 10 frames. The dense sampling scheme used a sampling rate that decayed with length, building on the intuition that a given frame in a short sequence carries relatively more information that a frame in a long GIF. The dense sampling scheme encoded all frames of GIFs with 10 or fewer frames, every other frame of GIFs with 11 to 50 frames, and at most 25 frames of longer GIFs, evenly distributed among the sequence of frames. Thus the maximum number of frames sampled in the dense scheme was 25. Next, we implemented a reshaping function that converted the JSON encoding file produced by the CNN into a (2048, num_frames) dimension vector encoding for each GIF. This vector served as the input to all models.

4.2 Baseline

We implemented two sentiment classification baselines. The first was a simple linear regression model that took the (2048, 1) dimensional encoding of the first frame of the GIF as input and gave 17 real-valued sentiment predictions as output. The linear regression model was trained with the same data as the other models. Improvements over this baseline would reveal the benefit of including the sequence information of subsequent GIF frames in making the prediction.

The second baseline was simply the class means for each of the 17 sentiment values. Because data were fairly clustered around the class means, a naive model could achieve relatively low error simply by always predicting the class means without varying between GIFs. Performing better than this baseline would indicate that the model had learned to use encoding features to explain variance in the sentiment scores.

4.3 Architectures

We explored two high-level architectures for the problem of sentiment classification, one using a fully connected network and one using an RNN. These are shown below in Figure 2.

4.3.1 Fully Connected with MaxPool

Building on Xu et al.'s [6] work with transfer learning of image sentiment, we hypothesized that passing the sentiment encoding to several fully connected layers may produce sufficient results. However, as the GIFs encodings consist of variable-length sequences of vectors, we first passed them through a MaxPool layer to collapse them to a single 2048 dimensional vector. The pooled vector was then fed through two or more fully connected layers with ReLU activation, culminating in a layer with 17 output units for each of the emotion classes. Because this model ignores the sequence ordering of the GIF frames, intuitively it can capture the presence of certain high-level features in the GIF (such as a smile or movement) but not action over time.

4.3.2 RNN

We also explore an RNN architecture similar to that used successfully for the captioning of GIFs by Li et al. [5]. This architecture fed the sequence of frame encodings to two or more LSTM layers of an RNN. Output from the final LSTM unit was sent to a fully connected layer with 17 outputs. This architecture better captures the sequence information encoded in the GIF.

4.3.3 Training

Both architectures were trained using mean-squared error for each of the 17 sentiment outputs. When used for evaluation, this error is averaged over all classes as shown in Equation (1) below, where b denotes the batch size and s the sentiment class.

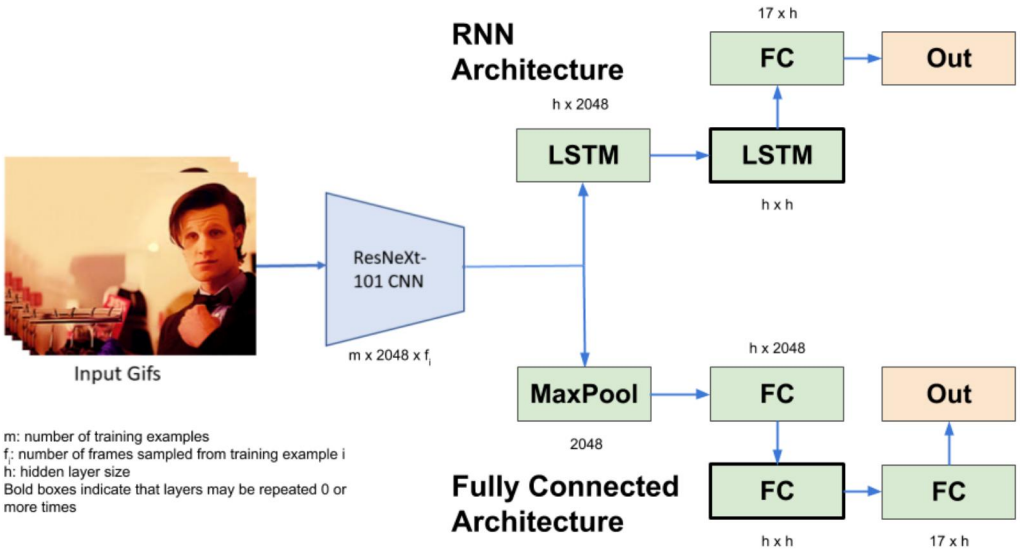

Figure 2: Illustration of two principal model architectures.

$$AverageMSE(\hat{Y}, Y) = \frac{1}{17b} \sum_{s=1}^{17} \sum_{i=1}^{b} (\hat{Y}_{i,s} - Y_{i,s})^2 \tag{1}$$

We used the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Learning rate and weight decay were tuned as part of the hyperparameter search process (see 5.2).

5 Experiments/Results/Discussion

We found that the best performing model was a two-layer RNN with 64 hidden units, batch size 256, learning rate of 0.000252, and dropout probability 0.1705, trained for 200 epochs. This achieved Root-MSE on the test set of 0.7610 and explained 40.8% of the variance in the data, marginally better than the other models. As can be seen from the train/dev loss in Figure 3, the high dropout rate during training helped the model avoid overfitting to the training set. Like the other best-performing models, it had a relatively small number of hidden units.

5.1 Training Set Sampling

Three training datasets were used: (1) Sparse Small, which sampled the training dataset with the sparse method described above, (2) Dense Small, which sampled the training dataset using dense sampling, and (3) Sparse Augmented, which sampled the augmented 43,995 GIF dataset using sparse sampling. (The augmented dataset was not sampled densely due to computational space and time constraints.) The sampling method on the dev dataset corresponded to that used on the training set. Thus models trained on Sparse Small or Sparse Augmented were tested using the validation set generated with sparse sampling. Performance for a given network was comparable across all three training datasets, with Dense Small yielding marginally better performance. The results below are thus reported for training and testing on Dense Small.

5.2 Hyperparameter Tuning

Hyperparameter search was performed for several model architectures: (1) Fully Connected with MaxPool (2 layers), (2) Fully connected with MaxPool (4 layers), (3) RNN (2 layers), (4) RNN (4 layers), and (5) RNN (8 layers). For each architecture, the model was trained for 20 iterations randomly sampling the learning rate on a log scale, number of hidden units (choosing among [32, 64, 128, 256, 512]), dropout (uniformly sampled between 0 and 50% drop probability), weight decay (sampled on a log scale), and batch size (choosing among [64, 128, 256]). The upper limit

Model	Train RMSE	Dev RMSE	Test RMSE	Test EV
Lin-Reg	0.07471	0.07849	0.08047	0.343
Mean Only	0.07336	0.07432	0.07611	0.406
FC-MaxPool-2	0.07336	0.07432	0.07612	0.406
FC-MaxPool-4	0.07362	0.07453	0.07627	0.406
RNN-2	**0.07120**	**0.07399**	**0.07610**	**0.408**
RNN-4	0.07341	0.07446	0.07613	0.406
RNN-8	0.07344	0.07431	0.07617	0.405

Table 1: Best performance on each model after tuning hyperparameters. Average Root Mean-Squared Error over all sentiment classes is reported, as well as explained variance on the test set.

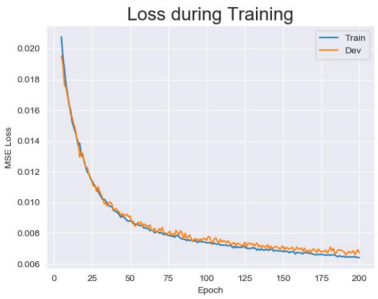

Figure 3: Train/Dev loss during one of the RNN-2 training runs.

on the batch size was set by memory constraints, while the lower limit was set by experimentation revealing no benefit to very small batch sizes such as 1. Dropout and weight decay each had a 50% probability of being set to 0. Based on preliminary experiments with the two-layer RNN, it was found that dev loss tended to plateau before 100 epochs (except in models with high dropout), so the performance of each model on the dev set was logged at 50 and 100 epochs. A few promising models from the hyperparameter search were retrained for more epochs. The best-performing models from hyperparameter tuning are reported in Table 1.

5.3 Error Analysis

5.3.1 Quantitative

The principal weakness of the model appears to be that it is too conservative, tending to make predictions very close to the mean value for each sentiment class. Indeed, the standard deviation of the predictions on the dev set by the best performing model was less than 0.02 for each class, while the true class standard deviations averaged 0.07. For this reason, the MSE by class is proportional to the class standard deviation. This helps explain why the model doesn't substantially outperform the Mean Only baseline.

5.3.2 Qualitative

A qualitative analysis of the 50 GIFs with the lowest average prediction error and the 50 GIFs with the highest average error reinforces the quantitative results that the model tends to hew too close to the class means. GIFs with low prediction error tended to lack strong emotions, or had ambiguous sentiment. Many of them consisted of a person with a neutral face staring off into the distance, lacking strong signals like smiles or frowns: Example 1, Example 2. In contrast, poorly predicted GIFs tended to have strong emotional features such as big smiles, tears, or wide eyes: Example 1, Example 2. These results suggest that the network is failing to distinguish key facial features as indicators of emotions. As discussed below, a different image encoder may address this problem.

6 Conclusion/Future Work

In summary, we found comparable performance among our models, with a two-layer RNN trained with dropout performing marginally better than the baseline of guessing the class means for each sentiment. This lack of improvement may be attributable either to the relatively small amount of data used or limitations of the CNN encoder.

Future work may explore using a different image encoder than ResNeXt-101 that is more suited to emotion detection, such as SentiBank[1] or another mid-level sentiment ontology. In particular, a network trained with facial features, such as that used by Jou et al.[4] may be beneficial. (ResNext-101 was primarily trained on videos of human actions and was chosen for its potential utility in the original project goal of multitask captioning and sentiment scoring of GIFs.) Retraining the final few layers of the encoder using transfer learning, as Xu et al.[6] applied to image sentiment classification, may also be of value. Finally, with more computational resources, the augmented dataset could be densely sampled, combining the benefits of more frames per training example and more examples.

7 Contributions

This project began as a collaboration with Chuma Kabaghe, who was unable to complete it due to illness. Chuma and Gordon originally worked together on multitask learning of GIF caption and sentiment data. They performed the literature search, proposal, and milestone together, as well as working together to modify the pretrained ResNext-101 CNN architecture to work on GIFs. Some elements of the final report text taken from previous milestones were written by Chuma. The remainder of the work focused on sentiment score prediction was done by Gordon, including obtaining and sampling the GIFGIF dataset, designing and implementing the baseline and model architectures, running experiments, and analyzing errors. Chuma is actively updating the GitHub for use in her own projects.

Thanks to TA Weini Yu for her advice and support throughout the project.

References

[1] Damian Borth et al. "Large-scale visual sentiment ontology and detectors using adjective noun pairs". In: *Proceedings of the 21st ACM international conference on Multimedia*. ACM. 2013, pp. 223–232.

[2] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. "Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?" In: *arXiv preprint* arXiv:1711.09577 (2017).

[3] Rongrong Ji et al. "Survey of visual sentiment prediction for social media analysis". In: *Frontiers of Computer Science* 10.4 (2016). DOI: 10.1007/s11704-016-5453-2.

[4] Brendan Jou, Subhabrata Bhattacharya, and Shih-Fu Chang. "Predicting Viewer Perceived Emotions in Animated GIFs". In: *ACM international conference on Multimedia*. 2014. DOI: 10.1145/2647868.2656408.

[5] Yuncheng Li et al. "TGIF: A New Dataset and Benchmark on Animated GIF Description". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[6] Can Xu et al. "Visual Sentiment Prediction with Deep Convolutional Neural Networks". In: *CoRR* abs/1411.5731 (2014).