Sentiment Analysis of Images using Transfer learning

Abhishek Vasudevan
Master's in Computer Science
University of Illinois at Chicago
avasud8@uic.edu

ABSTRACT
With the advent of social media platforms where people share images with little to no text, the task of sentiment analysis of images becomes important. In this paper, the task of visual sentiment analysis is performed using a transfer learning approach by employing a pretrained DenseNet-121 model on a large-scale dataset. This is evaluated with a VGG-19 model and through comparing accuracies of both models, it is proven that the DenseNet-121 model performs better than VGG-19.

General Terms
Machine learning, Measurement, Performance, Design

Keywords
Visual Sentiment Analysis, Transfer Learning, VGG-19, DenseNet-121

1. INTRODUCTION
Sentiment analysis is a widely explored topic in the field of machine learning. There is a plethora of methods available for finding out sentiment polarities of text. This is important as people frequently post on social media using text as the predominant form of communication. However, with the emergence of image sharing platforms such as Flickr and Instagram, the sentiment analysis of images becomes an important topic. This is because users usually post photos and videos on these sites with little to no description of the image i.e., the information about the post which can be extrapolated from the textual caption is limited. This provides the growing need to build models to understand images and use it to predict its sentiment polarity.

In this paper, we device a way to perform visual sentiment analysis by building a machine learning model and training it on a large dataset of images, sentiment polarity pairs (positive, negative, neutral).

2. RELATED WORK:
Early work focused on finding sentiment polarity of text. This has been widely researched and a plethora of methods have been devised for this process. Initially, linguistic features such as n-grams and Parts of Speech (POS) tags were employed to build machine learning models to learn polarity of text. These models include Naïve Bayes models, Support Vector Machines (SVM), deep neural networks (DNN). The most popular version of DNN would be the Long Short-Term Memory networks (LSTM) [1] which has proven to give top notch results for this task given a suitable dataset of text.

Visual Sentiment Analysis is new and is slowly gaining traction. One of the early works in this field is the understanding of human sentiments from images used an unsupervised framework proposed by Yuan et al. [2]. With the advent of CNNs for image feature extraction, sentiment analysis has been attempted using a Convolutional Neural Network where training dataset was created by asking annotators to label the images obtained from social media [3]. There has also been the use of Adjective-Noun pairs [4, 5] associated with images and a model trained to learn these mappings to predict sentiment polarity. The use of pretrained models like AlexNet was employed in a visual sentiment model [6] wherein the authors used a dataset called DeepSent, having 1269 images manually annotated by Amazon Mechanical Turk workers for training the model using transfer learning. The most important problem seems to be the lack of annotated data which is overcome in Vadicamo et al.'s paper [7] where a large scale of images are obtained and polarity labels to them are assigned in an ingenious way. They provide a transfer learning strategy which is the motivation behind our paper. The way the authors obtain the data is explained in detail in subsequent sections. It is worth mentioning at this point that the dataset provided by Vadicamo et al. [6] is several times larger in scale than any existing dataset used for training models for visual sentiment analysis. Naturally, their model achieves better results than all existing methods.
The motivation behind this paper is also that of [6] where the authors use transfer learning to train a VGG-19 model which is capable of recognizing objects in images and predict its sentiment polarity. Here, instead of a VGG-19 model, we use an even deeper and better model called DenseNet-121. The reason behind using a DenseNet-121 is straightforward: deeper models perform better in the task of object recognition. This can be used for transfer learning and make it understand sentiment polarities of images.

3. PROCESS OVERVIEW
Figure 1 shows the flow chart of the entire process starting from obtaining images to building a model to predict sentiment polarities.

We begin by obtaining the dataset released by the authors (who trained a VGG-19 to perform visual sentiment analysis). This is followed by extracting a subset of the dataset and preprocessing all images in that subset. The reason for choosing only a subset of the dataset rather the whole dataset is due to computational limitations as explained in subsequent sections. Next, a machine learning model pretrained to recognize objects in images, (DenseNet in our experiments) is retrained on this newly created dataset. Finally, we use a test dataset, details of which are explained later, to evaluate our trained model. The experimental results and comparison of VGG-19 and DenseNet-121 are provided. Finally, a discussion section and direction for future research are provided.

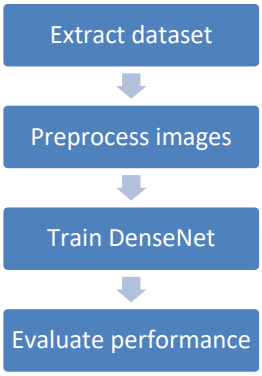

Figure 1. Flow chart of the entire process

4. DATASET

We use two distinct datasets for our training and testing purposes as explained below:

The dataset which we intended to use for training was fortunately made public and was downloadable online. This dataset is called the Balanced Twitter for Sentiment Analysis dataset (B-T4SA). It is a huge dataset consisting of equal number of examples for positive, negative and neutral images. Table 1 shows more details about the number of tweets and their corresponding images in the dataset. As we can see, the number of examples clearly should satisfy our training needs, for making the neural network model learn the distinctions between the three sentiment categories.

Sentiment	**T4SA**		**T4SA** w/o near-duplicates	**B-T4SA**
	(tweets)	*(images)*	*(images)*	*(images)*
Positive	371,341	501,037	372,904	156,862
Neutral	629,566	757,895	444,287	156,862
Negative	179,050	214,462	156,862	156,862
Sum	904,395	1,473,394	974,053	470,586

Table 1: Number of examples in the B-T4SA dataset

The way the authors obtained the B-T4SA dataset is as follows:

1) Obtain 1% of global tweets randomly from Twitter using Sample API.
2) Filter the extracted tweets such that the following are discarded: Retweets, tweets not in English, tweets less than 5 words in length and finally tweets not containing any image.
3) Use an LSTM-SVM model to classify the text of the tweet into one of three sentiment polarities.
4) Discard tweets whose text were classified with low confidence scores.
5) Assign images with the polarity label obtained from corresponding tweet text's label in step 3.

The flow diagram for this process is shown in figure 2. [8]

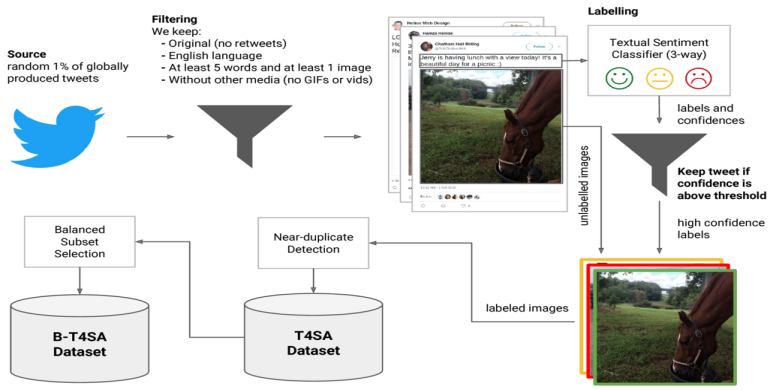

Figure 2: Flow process of creating B-T4SA dataset

The dataset has a satisfactory number of images, perhaps more than what is required to train a decent neural network model for polarity classification of images.

For testing the trained model, we use the Twitter Testing dataset which is the benchmark dataset for testing as this is used in several other papers as well. This dataset has 1269 images in total and labelled as either positive or negative by Amazon Mechanical Turk workers. The way each image was labelled is as follows: each image was labelled by 5 workers, where each worker would assign the image either positive or a negative label. As each image has the possibility of being assigned different labels since there are five different workers, the dataset is split into three categories as shown in table 2. Each image falls in the ≥ 3 category if more than 3 out of the 5 agree that it is positive/negative and similarly with the ≥ 4 and all 5 agree categories.

Sentiment	**Twitter Testing Dataset**		
	5 agree	**$\geq$ 4 agree**	**$\geq$ 3 agree**
Positive	581	689	769
Negative	301	427	500
Sum	882	1,116	1,269

Table 2: Categories and number of examples in Test dataset

For reporting results, we use the prediction accuracy of model on images and check the accuracy for images of all 3 categories i.e., 3 agree, 4 agree and all 5 agree.

5. METHOD USED

We start off by explaining transfer learning, the two different architectures which we wished to compare for visual sentiment analysis, preprocessing images and finally training the models.

5.1 Transfer learning

Transfer learning is becoming increasing popular as knowledge obtained in one domain can be used to retrain the same models and make it work with other similar domains. There are several pretrained models available online to the public thanks to the machine learning community. In particular, Tensorflow has made several pretrained models available online and all that is required is to import models and train it on our dataset of interest. The pretrained model used in the base paper is the VGG-19 while the model which we are aiming at training is the DenseNet121 models. Both of these are deep neural networks and have their weights trained on the ImageNet dataset [9]. Both of these models are trained to perform object recognition in images. By tweaking the architecture of these pre-trained models, they can be made to work with our domain of interest i.e., visual sentiment analysis. Rather than predicting objects in images, the models are made to predict sentiment polarity for images. The architecture and other specifics for these networks are explained in detail below:

5.2 VGG-19 architecture

The architecture of VGG-19 is shown in figure 3. As the name suggests, the VGG-19 has 19 layers built using convolutional and max pooling layers. It takes in input image of size 224x224x3 (RGB images) and outputs objects in the image. The convolution layers are of size 3x3 and the max pooling layers are of size 2x2 size. There are fully connected layers at the end. It has a top-1 error of 28.744% and top-5 accuracy of 10.012% on the ILSVRC dataset. [10].

Figure 3: VGG-19 architecture

5.3 DenseNet-121 architecture:

The DenseNet121 [11] is a much deeper architecture than VGG. The DenseNet architecture consists of a chain of Dense Block and transition layers as shown in figure 4**.** It has 121 layers and has a more complicated network architecture than VGG. In DenseNet models, every layer is connected with each other. In contrast to traditional neural nets with wider architectures, the DenseNet model is a narrow model and ensures maximum information/gradient flow between layers. DenseNet models also employ feature reuse. In this way, there is no need to learn redundant feature maps [12]. DenseNets do not sum the output feature maps of the layer with the incoming feature maps but concatenate them. This makes it different from traditional CNNs where the output is summed.

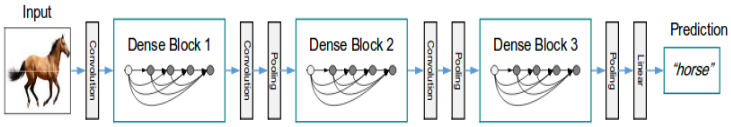

Figure 4: DenseNet-121 architecture

5.4 Comparison between VGG and DenseNet

The good thing about DenseNets is that they have fewer trainable parameters than VGG-19. This is clearly seen from the saved model file where the DenseNet model occupies less memory than the VGG model. DenseNet also is also much deeper than VGG ensuring better performance. Although, it is a deeper model, the training times for DenseNet and VGG are more or less the same owing to fewer trainable parameters in DenseNet. From the performance of the models on the ImageNet ILSVRC dataset, it can be seen that DenseNet has a lower top-1 and top-5 error percentage as shown in table 3.
All these advantages of DenseNet provide the motivation for using it instead of the VGG-19 model for the task of sentiment analysis of images, hoping that it would give better results.

Architecture	Top-1 error (%)	Top-5 error (%)
VGG-19	28.744	10.012
DenseNet-121	25.028	7.742

Table 3: Error (%) values between the 2 models

5.5 Preprocessing images

Regular preprocessing of images as for any neural network is employed here, however it makes sense to explain this in intricate details. The images are of variable size in the B-T4SA dataset. Therefore, they are normalized to size 224x224x3 during training time. All image pixel intensity values are normalized to be in the range 0 to 1 as this is the required value range for both VGG-19 and DenseNet-121 input images. Note that, some architectures require that the image be in a different range of normalized values.
The same procedure is adopted to images from the Twitter testing dataset. Once the images are brought into the format explained above, they are fed into the neural nets directly.

5.6 Confidence scores

The dataset by default has output labels of confidence scores for each of the 3 polarity classes, for each image. For any image, adding up the confidence scores for positive, negative and neutral classes would sum up to 1. Rather than using confidence scores, the polarity class having the confidence score is assigned a value 1 and the other polarity classes are assigned value 0. This is shown in the figure 5. Thus, each image has one output label of value 1 and other labels of value 0.

Image	Positive	Neutral	Negative
Image 1	1	0	0
Image 2	1	0	0
Image 3	0	1	0
Image N	...	...	...

Figure 5: Representation of preprocessed data

5.7 Training the neural nets

Once, the input images are preprocessed and the output labels are brought to the desired format, the model has to be designed and trained. For both the VGG-19 and DenseNet-121 architecture, the output layer is removed and replaced by the following set of layers:

1. Flatten layer
2. 30% dropout layer
3. Dense layer
4. Output SoftMax layer for 3 classes

The flatten layer is used to convert the 2D output from the final max-pooling layer to a 1D vector which is then processed by the dense layers. A dropout layer is added to prevent any overfitting during training. The pretrained VGG/DenseNet layers are frozen to prevent any changes in layer weights from the training process. The neural net outputs confidence scores at the SoftMax layer for each of the three sentiment polarities and the polarity with highest score is considered as the output prediction.

5.8 Testing the models

Once the model is trained, the Twitter Testing dataset is used to evaluate the model's performance. Since the model is trained to output one of 3 polarity classes (positive, negative and neutral) and the test dataset has only two output labels (positive, negative), the confidence score of positive and negative labels from model output is compared and the one having the higher score is considered as output.

6. EXPERIMENTAL SETUP

The training dataset (B-T4SA) has a huge dataset and due to lack of storage and compute power, only around 50,000 images of the original 470,586 (less than 10% of data) was used for training the DenseNet. The number of examples for each category is shown in table 4. It would be unfair to compare the performance of this DenseNet with that of the VGG-19 used in the base paper which was trained on the entire dataset. In order to make a fair comparison, we trained a VGG-19 model similar to the one reported in the base paper on this 10% dataset. Similar to the DenseNet, the output layer from VGG-19 was removed and replaced with the 4 additional layers mentioned in the previous section. Since both VGG and DenseNet accept images of size 224x224x3, the preprocessing of images for both these architectures are the same.

Category	Number of examples
Positive	15,441
Negative	14,887
Neutral	15,287

Table 4: Count for each sentiment class in the subset dataset of size 50k (It can be seen the number of examples is more or less balanced)

6.1 Training details

The same parameters are used for training both the VGG-19 and DenseNet models. Both models are trained for 20 epochs with an Adam optimizer. To prevent overfitting, L2 regularization was used in addition to the dropout layer mentioned in the previous section. The loss used was categorical cross entropy. A learning rate of 10^{-5} was used since if higher learning rates were used, both these models would rapidly overfit owing to their network complexity.

A small validation split of 5000 images was used to monitor the performance of the model as this would ensure that the model was not overfitting. The validation set was built by randomly selecting images from the training dataset.

6.2 Experimental results

After training both models by subjecting them to similar conditions as explained previously, it was found out that the DenseNet-121 outperformed the VGG-19 model. The evidence for this is shown in table 5 where the different agree metrics are shown for the two trained models and also the fine-tuned VGG-19 model reported in the base paper. We see a rough 3% increase in performance between the VGG-19 and DenseNet-121 models for all three "agree" metrics. Obviously, this is nowhere close to the metrics of the fine-tuned VGG which was trained on the complete dataset.

Model used	5 agree	>= 4 agree	>= 3 agree
VGG19 on 10% data	61.678	59.946	58.392
DenseNet121 on 10% data	64.62	62.00	60.75
VGG19 on whole dataset – fine tuned (base paper)	78.5	75.5	72.5

Table 5: Prediction accuracy on the Twitter testing dataset

The mean square error and mean absolute error from training the DenseNet-121 model is shown in figure 6. Naturally, with more epochs, the error decreases. Also, the accuracy from the validation split over 20 epochs is shown in figure 7. It starts off from 34%, all the way to around 45% on the B-T4SA dataset at the end of the 20th epoch.

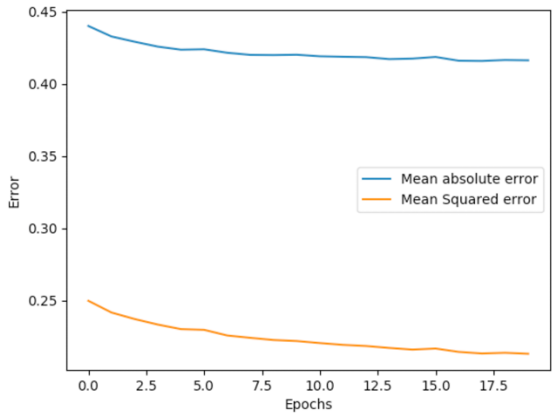

Figure 6: Error over 20 epochs

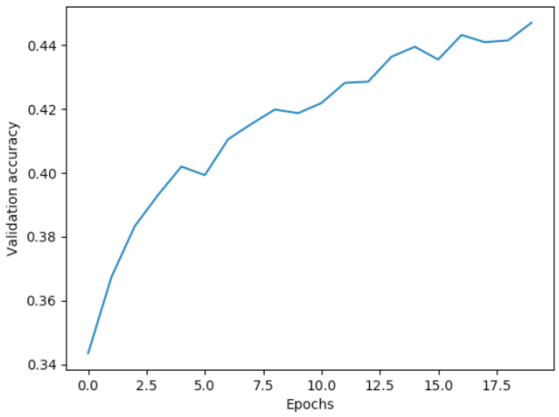

Figure 7: Validation accuracy over 20 epochs

6.3 Discussion

We see that just by using 10% of the training data provided, the DenseNet-121 performs better than the VGG-19 model. This is because that DenseNet is a more complex model with more convolutional and max-pooling layers. The vanishing gradient problem faced in such deep architectures is alleviated in this network. Furthermore, fine-tuning of the DenseNet layers which were frozen during training will definitely lead to a better performance, however this was not tried out in this experiment due to a lack of compute power.

Some of the images along with their predicted and ground-truth labels are shown in figure 8.

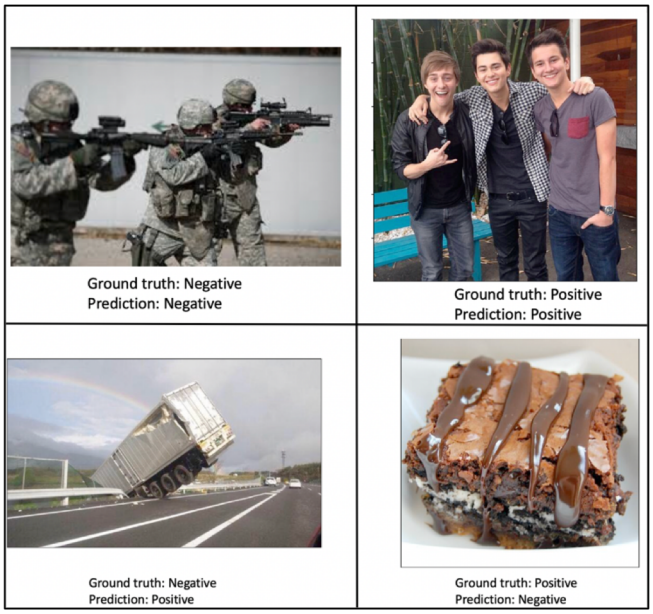

Figure 8: Example images and predictions

The model successfully predicted negative sentiment for an image showing soldiers which would normally invoke negative sentiments in humans. It also predicted positive sentiment for a group of people smiling. However, it failed to recognize negative sentiment for an image showing a crashed car. It also predicted negative sentiment for an image showing a cake which has been considered positive in the ground-truth. The model would have done a better job at predicting labels if it had access to the entire dataset. Also, by tweaking various other hyperparameters like increasing dropout and adding more dense layers, it is possible that the model would have learned more intricate features from images.

7. CHALLENGES FACED:

Initially, a ResNet-50 model was planned to be implemented and compared with the VGG model. However, even after changing various hyperparameters such as learning rate, adding several dense layers, regularization layers and dropout layers, the network would not learn anything as indicated by the fluctuating accuracy values during training for every epoch. This turned out to be because of the way batch normalization was implemented in Keras' pretrained ResNet model and the only way to fix it was to unfreeze all layers and train the model which was clearly undesirable as training all layers would be computationally too expensive. One another problem which was faced initially was the size of the training data. The dataset was roughly of size 63 GB and storing the entire dataset on the persistent disk provided by Google Cloud instance would be too costly. This was alleviated by using TensorFlow's data library and extracting only a part of the dataset and discarding the other images. The data library also allowed to load images in batches since loading 50,000 images all at once is beyond the memory

capacity of the virtual machine rented through Google Compute Engine.

8. CONCLUSION:

The paper compares the performance of transfer learning between two pretrained architectures on visual sentiment analysis: a VGG-19 and DenseNet-121 model. It was shown through experiments that the DenseNet model performed better than the VGG model by a small margin. Direction for further experimentation with these models has also been suggested. This future work is possible through obtaining more powerful compute power and the difference in performance between these two DNNs will be seen in an even larger scale. The code and model weights used in the experiments have been made public on Github. [13]

ACKNOWLEDGEMENTS

I thank the online deep learning community and my friends who gave a lot of guidance in training the deep learning models by suggesting various tweaks which I could perform in the hyperparameters. I also thank the authors of [6] who gave me access to the entire B-T4SA dataset which has helped me train the DNN models.

9. REFERENCES:

[1] D. Tang, B. Qin and T. Liu, "Document modelling with gated recurrent neural network for sentiment classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2015.

[2] J. Yuan, S. Mcdonough, Q. You and J. Luo, "Sentribute: Image Sentiment Analysis from a Mid-level Perspective," in *Proceedings of the Second International Workshop on Issues of Sentiment Discovery and Opinion Mining*, 2013.

[3] S. Jindal and S. Singh, "Image Sentiment Analysis using Deep Convolutional Neural Networks with Domain Specific Fine Tuning," in *International Conference on Information Processing (ICIP)* , 2015.

[4] B. Jou, T. Chen, N. Pappas, M. Redi, M. Topkara and S. Chang, "Visual affect around the world: A large-scale multilingual visual sentiment ontology," ACM, 2015.

[5] Y. Wang and B. Li, "Sentiment Analysis for Social Media Images," in *IEEE International Conference on Data Mining Workshop (ICDMW)*, 2015.

[6] V. Campos, B. Jou and X. Giro-i-Nieto, "From pixels to sentiment: Fine-tuning CNNs for visual sentiment prediction," *Image and Vision Computing,* 2017.

[7] Lucia Vadicamo et al., "Cross-Media Learning for Image Sentiment Analysis in the Wild," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017.

[8] Lucia Vadicamo et al., "T4SA dataset," [Online]. Available: http://www.t4sa.it/.

[9] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[10] A. Z. Karen Simonyan, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv preprint arXiv:1409.1556, 2014.

[11] G. Huang, Z. Liu, L. v. d. Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *Conference on Computer Vision and Pattern Recognition*, 2016.

[12] P. Ruiz, "Understanding and visualizing DenseNets," [Online]. Available: https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a.

[13] A. Vasudevan, "Github," [Online]. Available: https://github.com/abhishek-v/Image-sentiment-analysis.