Final Project - CS231n
Gaze Detection with CNNs for Linguistic Research

Rob Voigt
Stanford University
Linguistics Department
robvoigt@stanford.edu

Abstract

Eye gaze has been shown to be a factor of high psychological, social, and linguistic importance in human communication. Studies of gaze, however, often require expensive equipment or time-consuming annotation, that can limit the possibilities of what can be studied, at what scaled, and to what degree of generalizability.

In this final project I propose a pipeline for robust gaze detection in images of human faces, and train several convolutional neural networks to address the task, reporting accuracies of over 80% in nine-way directional classification, with sensical positional patterns of errors.

I also propose extensions to the domain video data, including a proposed frame-to-frame smoothing method. I report preliminary results on naturalistic spoken language data, which suggest that the models seem to have trouble generalizing, so I discuss difficulties in that domain.

1. Introduction

The body is an inherent part of the human communicative system, but the relationship of body movements to language is generally understudied in linguistics. One particular challenge lies in the difficulty of collecting accurate data: manual annotation of gestures and body movements is often time-intensive or otherwise expensive, so existing studies rely on very small datasets (as small as two minutes of video), limiting the generalizability of their findings.

My primary interest in computer vision is to develop methods for the automatic annotation of linguistically and socially relevant aspects of body movement so as to study their functions in communication. Our existing paper in this area used very simple image-difference vision features on a corpus of speech from YouTube to confirm the hypothesis that when speakers are acoustically excited – with higher and more variable pitch and loudness – they are concurrently moving their bodies more.[15] In this case, automatic methods not only allowed for the use of a larger corpus, but also for us to ask questions that would otherwise be impossible to ask, since unlike iconic gestures, overall body movement is perhaps impossible to annotate by hand.

A more subtle, but perhaps yet more socially meaningful bodily cue lies in eye gaze. Existing work in linguistics and psychology has shown that eye gaze often plays an important rule in infant and child language acquisition,[2, 7] turn-taking in dialogue,[3] and the disambiguation of referential expressions.[4]

Furthermore, gaze clearly has a strong social component. For example, subjects in an interview setting were more likely to rate interlocutors who had high levels of mutual eye contact as credible and attractive, and to associate gaze with intimacy and interpersonal similarity.[1] Moreover, fMRI neurological studies have demonstrated that gaze has a strong impact on the visual processing of faces [6] and differential processing of observed social interactions based on whether the agents' gaze was visible to subjects or masked.[11]

Like the question of overall body movement mentioned above, the study of gaze in linguistic communication is nearly infeasible without automatic methods. Therefore, in this paper I present work towards producing a system using convolutional neural networks (CNNs) capable of automatically annotating head-on videos with information about the speaker's gaze direction in linguistically meaningful ways.

I propose to then apply this system to existing corpora of videorecorded speakers giving naturalistic monologues to look for connections with body movement and acoustic prosody. In the current work I focus on checking the system's performance relative to a human standard annotation. However if it succeeds, in future work this system will allow us to ask new questions on a larger scale than previous linguistic research. For example, is diverted gaze consistently a signifier of disinterest (and thus low body movement and low pitch)? Is vertical gaze up or down indicative of positive or negative evaluations of the subject at hand?

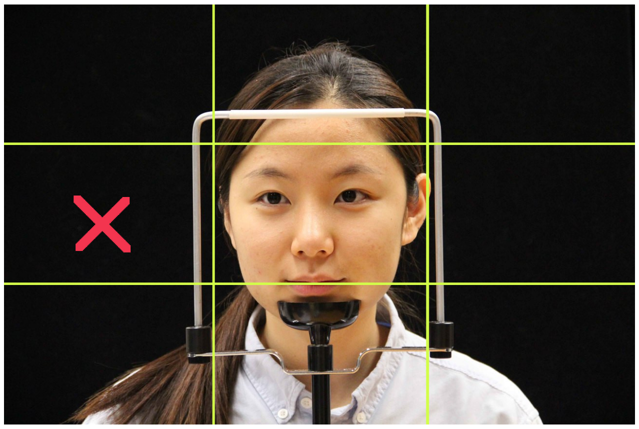
Figure 1. Visualization of the nine-way gaze classification task proposed in this paper.

2. Prior Work

The question of gaze detection has already been addressed in the literature. Some early work used color to generate masks for estimating head position with multi-layer perceptrons [12], but head position is more coarse than gaze and less relevant to my goals here. Much work has focused on HCI applications, which have the advantage of allowing for collecting speaker-specific training data. Existing studies demonstrate that the problem is a tractable one, however, and report, for example, accuracies within two degrees on webcam data have been reported using a two-layer feed-forward network,[13] and with PCA combined with a recurrent neural network.[5]

One of the most recent, large, and relevant data sets for this task is the Columbia Gaze Data Set. In the primary paper produced with this data set, the authors focused on gaze locking (that is, whether or not subjects were looking straight at the camera). They reported accuracies higher than human performance (an MCC of over 0.8) on the binary task of locked / not-locked, even at a distance of 18 meters from the subject.[14] In their work, they used a commercial face and fiducial point detector to extract eye regions, which they compress using principal component analysis (PCA) and multiple discriminant analysis (MDA) down to a six-dimensional subspace, within which they perform binary classification with a linear SVM.

3. Approach

In this work I propose a basic pipeline for gaze detection in static images making use of the Columbia Gaze Data Set. I then present preliminary results applying this system on existing linguistic video data.

In particular, I frame gaze detection as a relatively coarse nine-way classification task (see Figure 1), where eye gaze in an image is classified as being vertically centered, up, or down; and horizontally centered, left, or right. The labels used for this classification task, then, simply correspond to the product of these possible gaze directions.

3.1. Data

In this work I make use of the Columbia Gaze Data Set[1]. The dataset includes 5,880 images with 56 subjects taken at five horizontal head poses, with seven horizontal gaze directions (± 15, ± 10, ± 5, and 0) and three vertical gaze directions (± 10 and 0).

This data is still somewhat small for use with CNNs, but I consider several approaches to mitigate this factor. The data is also very clean, with participants in head-stabilizing harnesses and gaze targets placed at very regular intervals. An example of one image from the dataset may be seen in Figure 2. The cleanliness of that data may, however, have had negative impacts on the robustness of the downstream system, as I will discuss later.

3.2. Preprocessing

In this work I aim for the input to the system to be a relatively arbitrary image of a human face. Significant preprocessing is therefore necessary to provide consistent inputs to a downstream CNN capable of making decisions about gaze direction.

While some aforementioned prior work has addressed the issue of head pose, in this work I want to make a robust system that is capable of detecting gaze direction relative to the camera taking the image, regardless of the head pose of the subject. Such a system makes linguistic sense considering the general object of study is the perception of the listener, and whether the speaker is looking towards them or not; however, it adds significant noise to the system and difficulty to the modelling task.

Therefore, the preprocessing framework I settled on consists of three primary steps:

Face Detection

In this step, I use a pre-trained face detection module in DLIB[2], which employs Histogram of Gradients features in an image pyramid with a linear classifier and a sliding window detection scheme. This method is able to find the face in 99.2% of the images in the Columbia Gaze Data Set, and the remainder are not used in training or testing.

Landmark Finding

Given a face position, it is possible to make an estimate of the eye position by dividing the face into four regions

[1] `http://www.cs.columbia.edu/CAVE/databases/columbia_gaze/`

[2] `http://dlib.net`

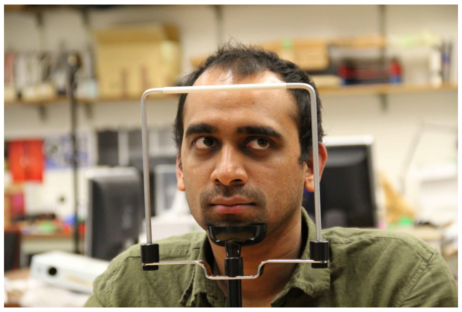
Figure 2. Original training image

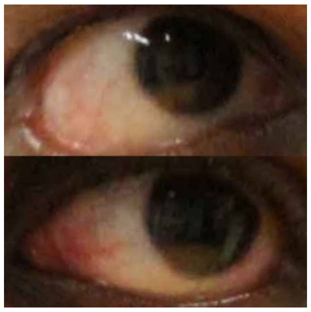
Figure 3. Input to the network after preprocessing

vertically and extracting the region found second-from-the-top. In initial experiments I used this method and used these regions directly as input to the classification models, but found performance to increase across the board by 5-10% when continuing forward with the somewhat more principled approach described below.

The authors in [14] note that they use a commercial face landmark detector to rectify eyes to a consistent representation; in this work, I again employ DLIB, which has an implementation of the model for shape prediction found in [9]. DLIB provides a model pre-trained on the iBUG 300-W face landmark dataset[3] to predict facial landmarks. I used this model to extracted predicted eye corner locations to be used in the next step.

Rectification

Using these four points per eye, I then employ OpenCV's[4] perspective transformation to transform each eye to a consistent 128 by 256 pixel space. I pad the eye corners by 16 pixels from the edge to allow for nearby facial textures and other potentially relevant information to not be lost, as well as to help buffer against potential noise introduced in the landmark detection process. This padding also has the potential to help downstream when doing crops for data augmentation.

[3]http://ibug.doc.ic.ac.uk/resources

[4]`http://opencv.org`

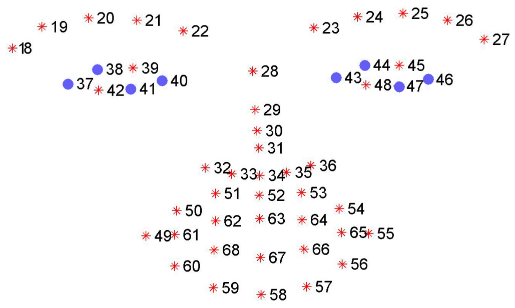

Figure 4. iBUG 300-W eye corner points used for rectification, marked in solid blue. 37, 38, 40, and 41 for the left eye; 43, 44, 46, and 47 for the right eye

4. Experiments

In these experiments I frame the problem as a nine-way classification task on the Columbia Gaze Data Set.

All horizontal labels in the data set are collapsed to be either left (all labels left of center), center, or right (all labels right of center). The vertical labels are already in three categories.

For baselines I used five-fold cross-validation since it was tractable and provides more consistent results; for the CNNs I divided the data set into 80%, 10%, and 10% train, validation, and testing splits, respectively.

4.1. Models

In this section I describe the various models tested.

Baselines

The models used for KNN, SVM, and SMX were all from the code provided with the homeworks for the class.

Most Common Class (MCC)

The most fundamental baseline, this reports guessing based on the most common class seen in the training data.

K Nearest Neighbors (KNN)

This baseline looks for the k nearest neighbors in the feature space, and picks as the correct class the one that receives a "majority vote" out of those neighbors. In this case I used $k = 5$.

Support Vector Machine (SVM)

This baseline uses the SVM loss objective to optimize a linear classifier; all reported results are using standard parameters trained for fifty iterations.

Softmax (SMX)

This baseline uses the Softmax loss objective to optimize a linear classifier; all reported results are using standard parameters trained for fifty iterations.

Nets

The three- and five-layer CNNs described below were trained using code provided with the homeworks for the class.

Three-layer CNN

This is a simple CNN using one convolutional layer followed by two affine layers and trained with a softmax loss. The convolutional layer is followed by a ReLU and a pooling layer, and the first affine layer is also followed by ReLU.

These models were trained with 32 filters of size 5 in the convolutional layer and 128 neurons in the hidden affine layer, and were trained from scratch with the entirety of the training data.

Five-layer CNN

This is a simple CNN that expands upon the previous model by beginning with three convolutional layers (each followed by ReLU and pooling).

These models were trained with 32, 32, and 64 filters, respectively, in the first three convolutional layers and 128 neurons in the hidden affine layer, and were trained from scratch with the entirety of the training data.

AlexNet

Since the dataset is reasonably large, but not large enough to train a truly full-scale convolutional network from scratch, transfer learning with fine-tuning of an existing model is an appropriate possible approach.

I used the Caffe[5] library to test this method, starting from the pre-trained Caffe reference model of AlexNet. [8, 10] AlexNet makes a particularly good model for this task, in part because its robustness to random crops may help when preprocessing is less than perfectly consistent.

4.2. Conditions and Data Augmentation

Baselines

All baseline models were trained both on the raw pixels of the images (resized to 64 by 64 for tractability) and on dimensionality-reduced feature vectors of length 500 computed using principal component analysis (PCA).

Nets

Initially the three-layer CNN was trained without data augmentation, and the best result in this condition is reported; however, I then implemented a data augmentation procedure to double the size of the dataset using random crops, tints, and contrast changes. Of course no flips were used, since positionality is centrally important to this task.

Hyperparameters were found using a manually-controlled coarse-to-fine random search in the space, code for which is attached.

[5]http://caffe.berkeleyvision.org/

BASELINES

Model	*Condition*	*Accuracy*
MCC		12.90%
KNN	Raw Pixels	14.20%
SVM		34.04%
Softmax		32.19%
KNN	PCA	36.68%
SVM	d=500	31.52%
Softmax		28.28%

Table 1. Baseline results; average over 5-fold cross-validation

NETS

Model	*Condition*	*Val*	*Test*
Three-layer		63.46%	60.18%
Three-layer	+Augmentation	74.54%	70.01%
Five-layer	+Augmentation	78.90%	71.63%
AlexNet	finetune SMX	55.45%	54.18%
AlexNet	+ all FCs	79.27%	76.18%
AlexNet	+ last conv	80.36%	79.27%

Table 2. Best achieved validation and test accuracies

Standard data augmentation in Caffe was used with AlexNet, but also with no flips/mirrors.

AlexNet was fine-tuned in three conditions: first, only the softmax layer (functionally using the net as a large preprocessing step to produce feature vectors); second, back through all fully-connected layers; and finally, back through the last convolutional layer before the fully-connected layers.

5. Results

The baseline results are shown in Table 1. These demonstrate that the robustness required for task is not possible to achieve with a straightforward linear classifier or KNN approach.

PCA for dimensionality reduction helped with the performance of KNN, but actually hurt for the SVM and Softmax classifiers; this seems related to some loss of direct positional information present in the raw pixel versions of those classifiers.

The performance of the convolutional nets, on the other hand, was much better, reaching validation accuracies above 80% and suggesting that perhaps the current approach is robust enough to apply to real-world data.

In particular, the fine-tuned AlexNet model worked the best, and the deeper back the fine-tuning went the better the overall performance. The five-layer model from scratch came near in performance, however, although at the cost of some significant overfitting that can be observed in the extracted features from its first convolutional layer, seen in Figure 7.

6. Linguistic Application

Eventually, my aim is to apply the above system to real-world linguistic data to extract interesting linguistic knowledge about how people integrate gaze into their conversational styles and strategies. Given the encouraging results above, I set above beginning to apply these models to some existing data and report my findings here.

6.1. Data

In other work I am currently collecting a corpus of YouTube monologues where individuals "vlog" and speak directly towards their screens. The videos are in some ways relatively controlled, because the backgrounds and cameras are often static, and once collected we know a priori that they contain one and only one speaker.

6.2. Video as Smoothing

The above models seem promising, but still not fully convincing as a result of performance above 90% might be; however, in the video context we can use subsequent frames as a form of smoothing, since we can reasonably expect that gaze does not in general change positions faster than once per several frames.

To that end I implemented a simple smoothing algorithm. First I extracted probabilities from succeeding frames of a video with the best-performing AlexNet model above (using the Caffe python API). Then, beyond just the argmax for a given frame (*raw*), I tried predicting the gaze direction based on a moving window of 5 frames, where the immediately following and preceeding frames are downweighted by a half, and the frames two back and two ahead are downweighted by a quarter (*smooth*).

6.3. Human Annotation

To check the robustness on real-world data of the models I trained, I implemented a simple program for human annotation of video. The program presents random frames from a video and asks the user to annotate the gaze direction to one of the nine categories shown above.

I used this program to annotate approximately 100 frames from each of three videos, and evaluated the model's automatic performance with reference to my annotations.

6.4. Findings

Table 3 presents the accuracies I found on real-world video. These are notably far below the accuracies I found with the controlled dataset used to train the models.

One video in particular was quite bad, with almost no matches between my annotations and the output of the system. Looking by hand, this is a video in which I perceived the speaker to be looking straight at the camera most of the time, but the system almost never made this judgement. I address this further below.

ACCURACY VS HUMAN

	raw	*smooth*
Video 1	44.04%	47.62%
Video 2	47.15%	48.21%
Video 2	4.08%	4.08%

Table 3. Accuracy compared to human annotation for individual frames taken from three YouTube videos, smoothed and unsmoothed

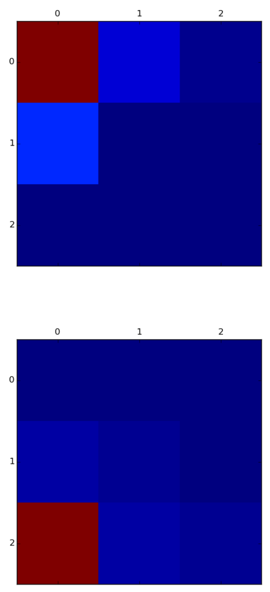

Figure 5. Confusion matrices on validation data for the upper-left and lower-left classes

7. Discussion

Overall, the models were generally able to learn and learn well from the training data; however, the size of the dataset was a notable problem. For all of the best-performing models, whether trained from scratch or fine-tuned, the training set accuracy was beginning to approach 100%, suggesting that overfitting was a serious issue.

Data augmentation with crops, tints, and contrasts helped this issue substantially, with a more than 10% boost to validation accuracy in my results. However, it may be the case that this particular dataset is too small and too closed-domain to be robust to real-world noise.

These models did not transfer very well to real-world linguistic data, as shown by comparison with human annotation. I think this discrepancy can be explained due to inavoidable overfitting on a small dataset, domain transfer issues, and changes with lighting and video quality.

To better observe what was actually happening, I plotted confusion matrices on the validation data with the five-layer CNN; almost all of them look very reasonable, with

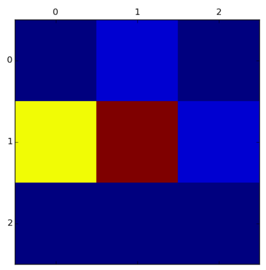

Figure 6. Confusion matrix for the (most difficult) center class

the probability mass loaded on the actual point of interest and errors surrounding it.

However, the worst-performing class for this model is actually the straight-on class. This may be due in part to the limited training data for that class: only around 300 examples in training and 20-30 examples each in the validation and test sets. This class also has the most neighbors to it.

These results suggest that perhaps a better model in the future might incorporate a loss function that explicitly values the center as the most important class above all others, rather than a broad classification objective which allows the other classes to overwhelm the central class in spite of its importance.

Overall, unfortunately, this system is not yet production-ready for real linguistic research, and a significant amount of additional tweaking, regularization, and perhaps even more naturalistic data collection will be necessary before it can truly be used.

References

[1] J. K. Burgoon, V. Manusov, P. Mineo, and J. L. Hale. Effects of gaze on hiring, credibility, attraction and relational message interpretation. *Journal of Nonverbal Behavior*, 9(3):133–146, 1985.

[2] T. Charman, S. Baron-Cohen, J. Swettenham, G. Baird, A. Cox, and A. Drew. Testing joint attention, imitation, and play as infancy precursors to language and theory of mind. *Cognitive development*, 15(4):481–498, 2000.

[3] C. Goodwin. Restarts, pauses, and the achievement of a state of mutual gaze at turn-beginning. *Sociological inquiry*, 50(3-4):272–302, 1980.

[4] J. E. Hanna and S. E. Brennan. Speakers eye gaze disambiguates referring expressions early during face-to-face conversation. *Journal of Memory and Language*, 57(4):596–615, 2007.

[5] T.-H. Huynh. A new eye gaze detection algorithm using pca features and recurrent neural networks. In *Computational Intelligence in Control and Automation (CICA), 2013 IEEE Symposium on*, pages 24–29. IEEE, 2013.

[6] R. J. Itier and M. Batty. Neural bases of eye and gaze processing: the core of social cognition. *Neuroscience & Biobehavioral Reviews*, 33(6):843–863, 2009.

[7] J. Jaffe, D. N. Stern, and J. C. Peery. conversational coupling of gaze behavior in prelinguistic human development. *Journal of Psycholinguistic Research*, 2(4):321–329, 1973.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[9] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1867–1874. IEEE, 2014.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] A. C. Pierno, C. Becchio, L. Turella, F. Tubaldi, and U. Castiello. Observing social interactions: the effect of gaze. *Social neuroscience*, 3(1):51–59, 2008.

[12] B. Schiele and A. Waibel. Gaze tracking based on face-color. In *Proceedings of the International Workshop on Automatic Face-and Gesture-Recognition*, pages 344–349. Citeseer, 1995.

[13] W. Sewell and O. Komogortsev. Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 3739–3744. ACM, 2010.

[14] B. Smith, Q. Yin, S. Feiner, and S. Nayar. Gaze Locking: Passive Eye Contact Detection for HumanObject Interaction. In *ACM Symposium on User Interface Software and Technology (UIST)*, pages 271–280, Oct 2013.

[15] R. Voigt, R. J. Podesva, and D. Jurafsky. Speaker movement correlates with prosodic indicators of engagement. In *Speech Prosody 7*, 2014.

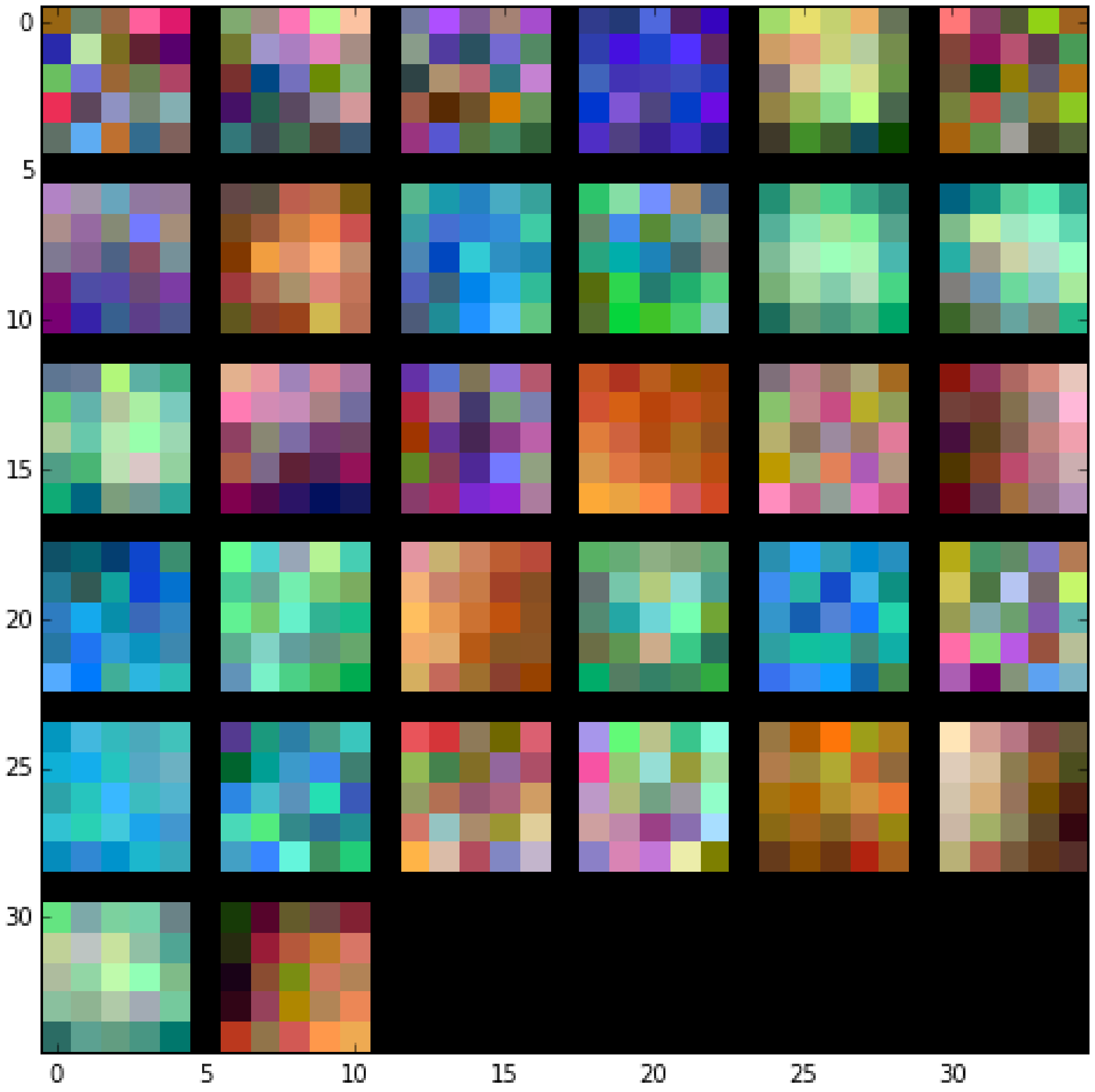

Figure 7. Extracted features in the first convolutional layer for the best-performing five-layer net.