Embedding Attention in CNNs for Improved Image Classification

Samuel Zureick

1 Introduction

Convolutional neural networks (CNNs) are a type of neural network (NN) primarily applied to tasks relating to visual data. In CNNs, a filter called a convolution takes many passes over the data, taking small squares of data, called the receptive field, at a time and, using the convolution filter, transforms the pixels in the receptive field into a single pixel to represent the contents of the receptive field. This process works well, but limitations are present. It is important to design a CNN that has a receptive field large enough to capture image context in deep layers in the CNN but small enough to capture important details in shallow layers of the CNN. By adding an attention layer to a convolutional neural network, a larger context window can be observed without drastically increasing the depth of the network, complexity of the model, or size of the receptive field. Using this method, along with Hyperband hyperparameter tuning, batch normalization and data augmentation, the testing accuracy of the CNN model presented in lab 2 for CIFAR10 image classifications with Keras can be increased to 85% without drastically increasing model complexity.

2 Background

In the field of robotics, deep neural networks (DNNs) have quickly overtaken many of the traditional methods used for a myriad of tasks. The task of speech recognition employed Hidden Markov Models for a long stretch of time, but the current state of the art for speech recognition employs DNNs[1]. The visual simultaneous localization and mapping (visual SLAM) problem has been studied in robotics for a long time, and recently elements of the typical visual SLAM pipeline have been interchanged with deep learning counterparts[2].

More recently in the field of deep neural networks (DNNs) has been a heightened focus on the attention mechanism, which allows a model to learn what features are most relevant to its current task. This is also relevant to the field of robot cognition, as it mirrors human attention mechanisms that allow us to find relevant details a large amount of data, visual[3] or otherwise. Wang et al. explore this idea by incorporating an attention network in their CNN, which allows for them to increase their classification performance[4] by learning the importance of specific features in the dataset.

CIFAR10 is an important benchmark for image classification as it requires a complex model with a high level of generalization to perform well; this is something that humans are very good at[5], and one of the aims of cognitive robotics is to create artifacts that work like humans so that we can learn more about humans and these artifacts[6].

3 Design

The model is trained for the task of multiclass image classification, which attempts to assign a descriptive label to a raw image taken as an input. This is an important task in the field of computer vision, and DNNS have shown success in applications such as facial recognition[7], self-driving vehicles[8], MRI analysis[9], and traffic flow analysis[10].

The details of the model architecture are shown in **figure 3.1**.

---

[1] Mustafa, M. K., Allen, T., & Appiah, K. (2019). A comparative review of dynamic neural networks and hidden Markov model methods for mobile on-device speech recognition. *Neural Computing and Applications*, *31*, 891-899.

[2] Li, R., Wang, S., & Gu, D. (2018). Ongoing evolution of visual SLAM from geometry to deep learning: Challenges and opportunities. *Cognitive Computation*, *10*, 875-889.

[3] Begum, M., & Karray, F. (2010). Visual attention for robotic cognition: A survey. *IEEE Transactions on Autonomous Mental Development*, *3*(1), 92-105.

[4] Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., ... & Tang, X. (2017). Residual attention network for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3156-3164).

[5] Ho-Phuoc, T. (2018). CIFAR10 to compare visual recognition performance between deep neural networks and humans. *arXiv preprint arXiv:1811.07270*.

[6] Morse, A. F., Herrera, C., Clowes, R., Montebelli, A., & Ziemke, T. (2011). The role of robotic modelling in cognitive science. *New ideas in psychology*, *29*(3), 312-324.

[7] Hassan, R. J., & Abdulazeez, A. M. (2021). Deep learning convolutional neural network for face recognition: A review. *International Journal of Science and Business*, *5*(2), 114-127.

[8] Do, T. D., Duong, M. T., Dang, Q. V., & Le, M. H. (2018, November). Real-time self-driving car navigation using deep neural network. In *2018 4th International Conference on Green Technology and Sustainable Development (GTSD)* (pp. 7-12). IEEE.

[9] Bernal, J., Kushibar, K., Asfaw, D. S., Valverde, S., Oliver, A., Martí, R., & Lladó, X. (2019). Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. *Artificial intelligence in medicine*, *95*, 64-81.

[10] Yi, H., Jung, H., & Bae, S. (2017, February). Deep neural networks for traffic flow prediction. In *2017 IEEE international conference on big data and smart computing (BigComp)* (pp. 328-331). IEEE.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 32, 32, 3)]	0	[]
conv2d (Conv2D)	(None, 32, 32, 64)	1792	['input_1[0][0]']
conv2d_1 (Conv2D)	(None, 32, 32, 64)	36928	['conv2d[0][0]']
batch_normalization (BatchNormalization)	(None, 32, 32, 64)	256	['conv2d_1[0][0]']
max_pooling2d (MaxPooling2D)	(None, 16, 16, 64)	0	['batch_normalization[0][0]']
attention (Attention)	(None, 16, 16, 64)	0	['max_pooling2d[0][0]', 'max_pooling2d[0][0]']
conv2d_2 (Conv2D)	(None, 16, 16, 256)	409856	['attention[0][0]']
conv2d_3 (Conv2D)	(None, 16, 16, 256)	1638656	['conv2d_2[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 16, 16, 256)	1024	['conv2d_3[0][0]']
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 256)	0	['batch_normalization_1[0][0]']
dropout_1 (Dropout)	(None, 5, 5, 256)	0	['max_pooling2d_1[0][0]']
flatten (Flatten)	(None, 6400)	0	['dropout_1[0][0]']
dense (Dense)	(None, 512)	3277312	['flatten[0][0]']
dropout_2 (Dropout)	(None, 512)	0	['dense[0][0]']
dense_1 (Dense)	(None, 10)	5130	['dropout_2[0][0]']

Total params: 5,370,954
Trainable params: 5,370,314
Non-trainable params: 640

Figure 3.1: Model architecture for CNN with attention

The model is designed using the Keras functional API, as it allows for more complex models to be created. The first two layers are 2D convolutions with 64 filters each, a stride of (1,1), kernel size of 3, equal padding, and RELU activation function. Following this is a batch normalization layer, which normalizes the outputs of the preceding convolutional layers to prevent overfitting and stabilize the learning process[11]. After batch normalization is a MaxPooling layer with a size of (2,2), which is used to down sample the output from the convolutional layers in order to capture more general features in subsequent layers of the network[12], followed by a dropout layer with a rate of .25, which further aids model generalization.

Following this first block, an attention layer is added, using the MaxPooling layer output as its query and key inputs. This layer calculates the Luong-style attention[13] of the result of the first complete convolutional block, which has been shown effective for image classification tasks[14] by Bello et al., and acts to learn the importance of the features down sampled from the convolutions.

After the attention layer in the model architecture is another block like the first, but with a higher filter count and larger kernel size in the convolutional layers and a larger max pooling size.

Finally, the layers are flattened, passed through a dense neural network of size 512, passed through a dropout layer, and then passed to a final classifying layer with a softmax activation function.

The model is compiled with AdamW as the optimizer and loss function of categorical crossentropy.

AdamW was chosen as the optimizer over SGD because although SGD can achieve similar performance to AdamW with less compute, Kumar et al. demonstrate that AdamW with a fine-tuned model can perform significantly better[15] for computer vision tasks.

The hyperparameters chosen for this model were all determined through the Keras Hyperband Tuner, a hyperparameter optimization tool. The hyperband tuner works by compiling a random array of models with different hyperparameter selections and partially training them, keeping those that perform best according to the user specified objective, in this case validation accuracy, and continuing to train these best performing models against one another until one set of hyperparameters emerges as the optimal. Hyperband is a type of random search, and it does converge to the optimal hyperparameter selection, but uses explore-exploit strategies to do this much quicker than a true random search[9]. In this case, the hyperparameters input into the tuner were the filter and kernel sizes of the 2D convolutional layers, as well as the size of the pooling layers and dense neural network. Hyperband tuning can be parallelized, providing a much more efficient[16] way to tune hyperparameters compared to typical Bayesian methods.

[11] Ogundokun, R. O., Maskeliunas, R., Misra, S., & Damaševičius, R. (2022, July). Improved CNN Based on Batch Normalization and Adam Optimizer. In *Computational Science and Its Applications–ICCSA 2022 Workshops: Malaga, Spain, July 4–7, 2022, Proceedings, Part V* (pp. 593-604). Cham: Springer International Publishing.

[12] Scherer, D., Müller, A., & Behnke, S. (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In *Artificial Neural Networks–ICANN 2010: 20th International Conference, Thessaloniki, Greece, September 15-18, 2010, Proceedings, Part III 20* (pp. 92-101). Springer Berlin Heidelberg.

[13] Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

[14] Bello, I., Zoph, B., Vaswani, A., Shlens, J., & Le, Q. V. (2019). Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 3286-3295).

[15] Kumar, A., Shen, R., Bubeck, S., & Gunasekar, S. (2022). How to Fine-Tune Vision Models with SGD. *arXiv preprint arXiv:2211.09359*.

[16] Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2017). Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, *18*(1), 6765-6816.

Figure 3.2 shows the setup involved for hyperband tuning.

```
N_EPOCH = 40 # 40 bigger network will benefit from extra training epochs

inputs=keras.Input((IMG_ROWS, IMG_COLS, IMG_CHANNELS))

def builder(hp):
  f1 = hp.Int('f1', min_value = 16, max_value = 512, step=16)
  ks1 = hp.Int('ks1', min_value=3, max_value=5, step=2)
  ks2 = hp.Int('ks2', min_value=3, max_value=5, step=2)
  ps1 = hp.Int('ps1', min_value=2, max_value=3, step=1)
  c1 = Conv2D(64, kernel_size=ks1, padding='same', activation='relu')(inputs)
  c2 = Conv2D(54, kernel_size=ks2, padding='same', activation='relu')(c1)
  n1 = keras.layers.BatchNormalization()(c2)
  p1 = MaxPooling2D(pool_size=(ps1, ps1))(n1)
  d1 = Dropout(rate=.1)(p1)

  a = keras.layers.Attention()([p2,p2])
  f2 = hp.Int('f2', min_value=16, max_value=512, step=16)
  ks3= hp.Int('ks3', min_value=3, max_value=5, step=2)
  ks4 = hp.Int('ks4', min_value=3, max_value=5, step=2)
  ps2 = hp.Int('ps2', min_value=2, max_value=3, step=1)
  c3 = Conv2D(f2, kernel_size=ks3, padding='same', activation='relu')(d1)
  c4 = Conv2D(f2, kernel_size=ks4, padding='same', activation='relu')(c3)
  n2 = keras.layers.BatchNormalization()(c4)
  p2 = MaxPooling2D(pool_size=(ps2, ps2))(n2)
  d2 = Dropout(rate = 0.25)(p2)

  f = Flatten()(d2)
  hp_units = hp.Int('hpu', min_value=32, max_value=1024, step=32)
  d3 = Dense(units=hp_units, activation='relu')(f)
  d4 = Dropout(rate = .5)(d3)
  d5 = Dense(N_CLASSES,activation="softmax")(d4)
  model = keras.Model(inputs = inputs, outputs = d5)
  model.compile(loss='categorical_crossentropy', optimizer=OPTIM, metrics=['accuracy'])
  return model

tuner = kt.Hyperband(builder, objective='val_accuracy', max_epochs = 30, factor=3)

stop_early = keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)
model.summary()
tuner.search(input_X_train, output_Y_train, validation_split=.2, callbacks=[stop_early])
```

Figure 3.2: Design of hyperband tuning

Supporting the network architecture is data augmentation, which introduces slight transformations to the dataset in order to artificially create more training examples and train a more generalized model. Because DNNs trained on more data typically perform better regarding accuracy compared to the same model trained on less data[17], creating more examples with training augmentation can improve model accuracy and help avoid the problem of overfitting[18].

4 Results

The model resulting from hyperband hyperparameter selection was composed of 5,370,954 total parameters. The model is relatively simple and can generalize well thanks to the attention mechanism and data augmentation.

The model was trained for 45 epochs on 50000 training examples, with 10000 of these set aside for validation, and was tested on 10000 examples.

Metric	Score
Loss (Categorical Crossentropy)	.467
Accuracy	.854
Precision	.900
Recall	.823
F1	.855

Figure 4.1: Results of evaluation of final model over 10000 test samples

Figures 4.2 and **4.3** show the training and validation accuracy and loss over 45 epochs, respectively.

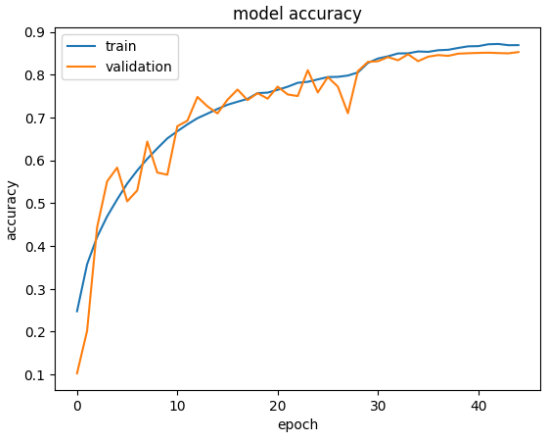

Figure 4.2: Training and validation accuracy over 45 epochs

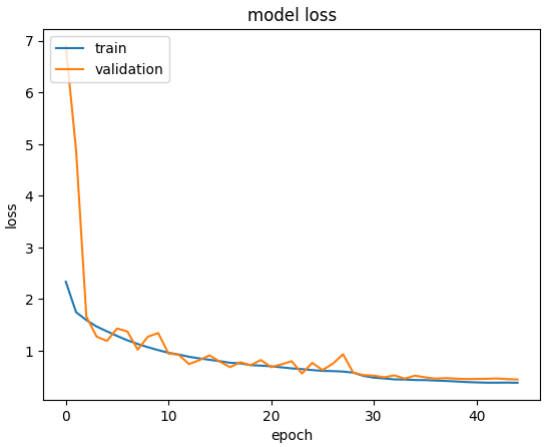

Figure 4.3: Training and validation loss over 45 epochs

5 Discussion

The testing results show an excellent increase in performance compared to previous DNN

[17] Luo, C., Li, X., Wang, L., He, J., Li, D., & Zhou, J. (2018, November). How does the data set affect cnn-based image classification performance?. In *2018 5th international conference on systems and informatics (ICSAI)* (pp. 361-366). IEEE.

[18] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.

models introduced in the labs. The model, however, is significantly more complex than those in the labs, takes longer to train, and requires much more compute.

The validation and test results indicate that the model can generalize appropriately- test, train, and validation accuracy and loss are all similar, showing that the model is not overfitting the data. Accuracy could be increased by increasing the model complexity and the size of the training data.

This result reflects one of the disadvantages of DNNs for cognitive robotics- in order to build and train a model with high accuracy, massive amounts of data are required. Data augmentation can help a model generalize and increase test accuracy[19], but models trained on relatively small datasets struggle to match the performance of those trained on larger datasets, even with image augmentation[20].

Computer vision tasks such as facial recognition have already demonstrated harmful consequences when biased models are used for tasks such as policing and surveillance[21]. One way to improve these struggling models is to collect more data, but often the techniques invite in a plethora of data privacy and safety concerns[22].

The use of big data also incurs high costs related to data acquisition, storage and security[23]. Additionally, data storage centers are large attack targets, and the fines, legal fees, and security costs associated with data breaches can be astronomical; Equifax's data breach cost them $575 Million in settlement fees alone[24], with costs relating to the breach reaching over $1.35 Billion as reported by Equifax in 2019[25].

One way to expand this implementation would be to introduce more complex data augmentation techniques. In this project, only geometric transformations have been used, but kernel filters, color distortion, and information deletion could be employed, which have been shown to increase model robustness[26] and increased accuracy[27].

Another approach to increase generalization in training with Data Augmentation Generative Adversarial Networks (DAGANs), which uses an adversarial neural network to generate novel input that is valid training data that explores underexploited features in the original data[28].

6 Conclusion

By increasing the model complexity for a CNN for image classification, test accuracy for the CIFAR-10 dataset can be improved. Embedded attention supplements the convolution and pooling, allowing for the most important image features to be extracted from images. Data augmentation aids in the generalizability of the model, and paired with dropout layers, helps to decrease model overfitting.

Hyperband tuning is shown to be effective at selecting an optimal model containing a wide array of hyperparameters in less time and with less compute than typical grid search or Bayesian optimization techniques.

While computer vision models are still not as good at generalization and abstraction as humans in some domains, there are many important applications of DNNs in computer vision that are worth developing and pursuing, given safe handling of the large amounts of data required to train an affective model.

[19] Chen, P., Liu, S., Zhao, H., & Jia, J. (2020). Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*.
[20] Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., & Beyer, L. (2021). How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*.
[21] Dauvergne, P. (2022). Facial recognition technology for policing and surveillance in the Global South: a call for bans. *Third World Quarterly*, *43*(9), 2325-2335.
[22] Scharre, P. (2019). Killer apps: The real dangers of an AI arms race. *Foreign Aff.*, *98*, 135.
[23] Chen, X. W., & Lin, X. (2014). Big data deep learning: challenges and perspectives. *IEEE access*, *2*, 514-525.
[24] https://www.ftc.gov/news-events/news/press-releases/2019/07/equifax-pay-575-million-part-settlement-ftc-cfpb-states-related-2017-data-breach
[25] https://www.bankinfosecurity.com/equifaxs-data-breach-costs-hit-14-billion-a-12473
[26] Naveed, H., Anwar, S., Hayat, M., Javed, K., & Mian, A. (2021). Survey: Image mixing and deleting for data augmentation. *arXiv preprint arXiv:2106.07085*.
[27] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.
[28] Antoniou, A., Storkey, A., & Edwards, H. (2017). Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*.