Parallel Visual Attention Encoder for Model Improvement

Fernando Martinez
Computer and Information Sciences Department
Fordham University
113 W 60th Street, New York, NY 10023
fmartinezlopez@fordham.edu

***Abstract*—In this project, we propose a convolutional attention module that can be incorporated alongside state-of-the-art convolutional neural network (CNN) architectures in parallel. The use of convolutional encoders allows for efficient feature extraction and dimensionality reduction. At the same time, a visual attention module enables the model to focus on the most relevant features for a given task. We developed an architecture that takes EfficientNetB2 and combines its output with another branch trained in parallel, encodes the inputs and performs visual attention on the latent block. For this purpose, we used a hybrid approach that simultaneously passes the encoding output through two attention modules, combines their results, and then merges the output with the base model outcome for further processing.**

The developed model performance presents a validation accuracy of 89.52% working with the Stanford Dog Dataset and 82.99% with the Tsinghua dataset, whereas EfficientNetB2 on its own presents 89.14% and 82.02% respectively.

***Index Terms*—ConvNets, Encoder, Attention, Deep Learning**

I. Introduction

Image classification problems tend to bring multiple challenges when it comes to working with day-to-day images captured either by our phones or by a professional camera; because the photos we provide to our model are not perfectly focused or they were not taken in the perfect scenario, there is too much noise, or there is something in the middle of the object. Over the last few years, convolutional neural networks (CNNs) have greatly contributed to multiple disciplines and applications due to their ability to extract relevant visual representations. [1] [2]. One of the key properties of CNNs is their ability to learn local, translation-invariant features from the input data, which makes them effective at recognizing patterns and objects in images despite variations in position and appearance.

As humans, visual attention is a key aspect of human perception, allowing us to focus on relevant information in our environment while ignoring irrelevant distractions. In recent years, there has been increasing interest in developing artificial systems that can also exhibit visual attention, particularly in the context of machine learning and computer vision. Similar to what we do with human vision, these techniques aim for our models to ignore the noise around objects and concentrate their attention on what is relevant to the visual task.

This paper proposes a convolutional encoder module with a visual attention latent block that can be incorporated alongside state-of-the-art CNN architectures in parallel. The use of a convolutional encoder allows for efficient feature extraction and dimensionality reduction, while the visual attention module enables the model to focus on the most relevant features for a given task. By combining these two approaches, our module is able to achieve improved performance on the tested image classification deep learning architecture.

Our experimental results demonstrate the effectiveness of the proposed architecture, achieving state-of-the-art performance on the two benchmark datasets. In addition, we provide a thorough analysis of the model's behavior and ability to learn relevant features, highlighting this hybrid approach's potential for future research.

II. Related Work

There has been a significant amount of research in the field of attention mechanisms for neural networks. Some early examples include the Attention Mechanism [3] and the Neural Attention Model for Abstract Sentence Summarization [4]. These models demonstrate the effectiveness of using attention to selectively focus on certain parts of an input to make more accurate predictions for natural language processing ends.

More recently, there has been a focus on incorporating attention mechanisms into CNNs for image classification tasks. One example is the Squeeze-and-Excitation (SE) Network [5], which introduces a channel-wise attention mechanism to allow the network to adaptively recalibrate channel-wise feature responses. Another example that motives the approach developed in this project is the Convolutional Block Attention Module (CBAM) [6], which is also a CNN-based attention mechanism, specifically designed for image classification tasks. CBAM introduces both channel-wise and spatial attention mechanisms, allowing the network to attend to both the channels and spatial locations of the input features. By doing so, CBAM is able to capture both global and local dependencies in the input, leading to improved performance on various image classification benchmarks.

Additionally, this year brought another novel architecture based on visual attention, Visual Attention Network (VAN) [7]. The authors demonstrated how VANs outperform other cutting-edge architectures in multiple visual tasks such as image classification, semantic segmentation, object detection, pose estimation, etc. In this paper, they proposed a new attention mechanism called large kernel attention (LKA) which

extracts the advantages of convolution, and self-attention [8] [9]. The authors of LKA defined the mechanism as an adaptative selection process capable of selecting the discriminative features and automatically ignoring noisy responses based on the input features [7].

III. Data used for this project

A. Training and validation data

To benchmark our model's performance, we used a total of two (2) datasets. Both used datasets in this project have a random train-validation split ratio of 80%-20%, respectively.

1) Stanford Dogs Dataset: The Stanford Dog Dataset contains images of 120 different dog breeds. This dataset has a total of 20,580 images of dogs from all around the world [10]. After the split, the dataset is divided into two groups, training and validation sets, which contain 16,464 and 4,116, respectively.

Fig. 1: Stanford Dogs Dataset Sample Images

2) Tsinghua Dogs Dataset: The Tsinghua Dog Dataset contains images of 130 different dog breeds. This dataset has a total of 70,428 images. Each dog breed class contains at least 200 images and a maximum of 7,449 images. [11]. After the split, the dataset is divided into two groups, training and validation sets, which contain 56,346 and 14,086 respectively. Compared to the Stanford dataset, this dataset contains more noisy general data but also includes a deceptive new class named "Chinese Street Dogs" that includes dog images from the street and could be from any dog breed or even mixed breeds.

Fig. 2: Tsinghua Dogs Dataset Sample Images

B. Real dog images captured by relatives

As we will explore later, we collected some images from relatives and friends and tested our best model using a web application developed for this project. Figure 3.

Fig. 3: Relatives' Dogs Sample Images

IV. Methods

In this section, we first give a deeper introduction to the concepts applied to develop our parallel visual attention encoder to enhance models' performance. We then present our new approach, which combines CBAM [6] and LKA [7] into an encoder module and can be used with other state-of-the-art deep learning architectures.

A. Convolutional Block Attention Module (CBAM)

CBAM is a type of visual attention mechanism designed to improve the performance of CNNs by adaptively weighting the channel and the spatial dimensions [6]. Both channel and spatial attention submodules use global average pooling and max pooling. After retrieving the global max and global average, the channel attention submodule passes in parallel both results through a sequence of two (2) dense layers, then sum the two (2) parallel branches and applies the sigmoid activation function to the output. The spatial attention submodule concatenates its input altogether, applies a convolutional layer with a large kernel and sigmoid as its activation function, and then multiplies the input by the layer's output.

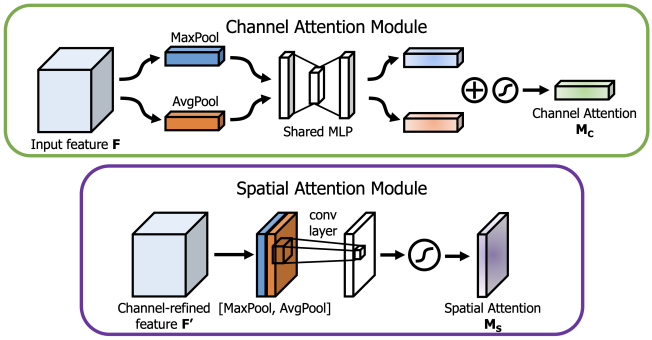

Fig. 4: CBAM: Diagram of each attention sub-module. *Image reproduced from* [6]

The authors determined during their experiments that the order these two submodules are arranged affects the overall performance. The best arrangement is Channel-Spatial.

B. Large Kernel Attention (LKA)

The authors of VAN [7] developed the large kernel attention (LKA) module to generate attention maps that highlight the

importance of different pixels in an image. This module was designed to address the problem of computational overhead and the large number of parameters required by self-attention modules [8]. LKA decomposes a large kernel convolution into three parts: a local spatial convolution (depth-wise convolution), a long-range spatial convolution (depth-wise dilated convolution), and a channel convolution (1x1 convolution). This decomposition allows LKA to efficiently capture long-range relationships, determine the importance of a point, and generate attention maps.

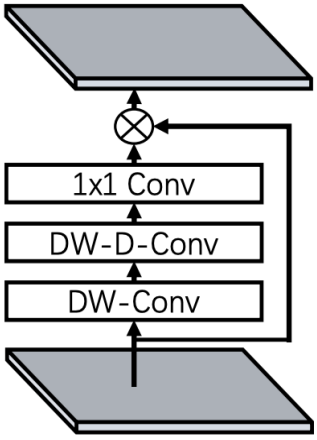

Fig. 5: Large kernel attention (LKA) structure. *Image reproduced from* [7]

C. Parallel Visual Attention Encoder for Model Improvement - FerNet

Our method proposes the use of a neural network architecture based on EfficientNetB2 (fine-tuned) [12], adding a branch that performs visual attention by combining LKA and the CBAM modules. FerNet introduces a sequence of layers that work as an encoder, similar to the ones we find in AutoEncoders. These layers are responsible for dimensionality reduction and the efficient extraction of features taking the input in parallel alongside the base model (EfficientNet). Instead of a regular latent layer, FerNet introduces the LKA and CBAM modules and executes them simultaneously. After extracting the most relevant features and running the attention modules, the model passes the submodules' outputs through two (2) convolutional layers of the same size. Then it merges them altogether (sum both submodules' outputs), processes the sum through another convolutional layer (that uses GELU [13] as the activation function), executes an average pooling for both objects, and, finally, flattens all the pixels.

Subsequently, the proposed model concatenates the previous result with the base model resulting tensor and operates them through a regular dense neural network until generating the classification output. A significant advantage of our approach is that we are not sacrificing computation to get the improvement we are looking for since the encoder is responsible for reducing the required operations. Both attention modules efficiently detect what pixels are the most relevant, ignoring possible noise in our inputs. A diagram of our model is presented in figure 6.

The setting we use for each layer is as follows:

TABLE I: FerNet Configuration

Layers	Output Shape	Module	Connected To
Rescaling	(260, 260, 3)	Encoder	Model's input
ConvEncoderAtt1	(258, 258, 32)	Encoder	Rescaling
AvgPooling	(129, 129, 32)	Encoder	ConvEncoderAtt1
ConvEncoderAtt2	(127, 127, 16)	Encoder	AvgPooling
AvgPooling2	(63, 63, 16)	Encoder	ConvEncoderAtt2
Lambda-MaxPool	(1, 1, 16)	CBAM-Channel	AvgPooling2
Lambda-AvgPool	(1, 1, 16)	CBAM-Channel	AvgPooling2
ChannelMaxConv1	(1, 1, 16)	CBAM-Channel	Lambda-MaxPool
ChannelMaxConv2	(1, 1, 16)	CBAM-Channel	ChannelAttMaxConv1
ChannelAvgConv1	(1, 1, 16)	CBAM-Channel	Lambda-AvgPool
ChannelAvgConv2	(1, 1, 16)	CBAM-Channel	ChannelAttAvgConv1
Add1	(1, 1, 16)	CBAM-Channel	AttAvgConv2-MaxConv2
Multiply1	(63, 63, 16)	CBAM-Channel	AvgPooling2-Add1
Lambda-MaxPool2	(63, 63, 1)	CBAM-Spatial	Multiply1
Lambda-AvgPool2	(63, 63, 1)	CBAM-Spatial	Multiply1
Concatenate1	(63, 63, 2)	CBAM-Spatial	LabmbdaAvgMax2
SpatialConv1	(63, 63, 1)	CBAM-Spatial	Concatenate1
Multiply2	(63, 63, 16)	CBAM-Spatial	SpatialConv2-Multiply1
SpatialConv2	(63, 63, 4)	CBAM-Spatial	Multiply2
ProjConv1	(63,63, 12)	LKA	AvgPooling2
BatchNorm	(63,63, 12)	LKA	ProjConv1
DWProjSPatial1	(63,63, 12)	LKA	BatchNorm
DWProjSPatial2	(63,63, 12)	LKA	DWProjSPatial1
ProjConv2	(63,63, 4)	LKA	DWProjSPatial2
ProjConv3	(63,63, 12)	LKA	ProjConv2
Add2	(63,63, 12)	LKA	ProjConv3-ProjConv1
Conv1	(63,63, 4)	Add2	Add2
Add3	(63,63, 4)	AttLatent	SpatialConv2-Conv1
LantentOutput	(30,30, 4)	AttLatent	Add3
AvgPooling3	(15,15, 4)	AttLatent	LantentOutput
Flatten	(None, 900)	AttLatent	AvgPooling3

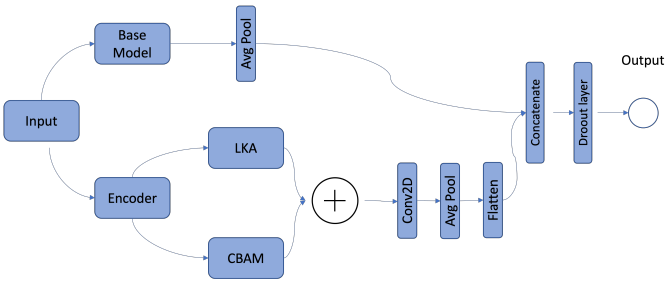

Fig. 6: FerNet Structure

D. Techniques used in this project

We used distributed parallel training to train our models faster, given that the Kaggle Environment provides two (2) GPU NVIDIA T4, improving our training time in half the time. By default, the environment is set to use only one GPU. In order to improve our model's performance, we implemented fine-tuning by unfreezing a portion of the base model's layers after a certain number of epochs of feature extraction. For the Stanford dataset, we ran 15 epochs of feature extraction with the base model layers frozen, then unfroze 10% of the layers

and ran an additional 15 epochs of fine-tuning. We used a similar approach for the Tsinghua dataset with ten (10) epochs of feature extraction and ten (10) epochs of fine-tuning. During the fine-tuning stage, we reduced the learning rate as a best practice [14]. Moreover, each model was compiled using the Adam optimizer [15] and cross-entropy for the loss function. The applied loss function is not the vanilla version; instead, we applied label smoothing, which relaxes each label's confidence to avoid our model being excessively convinced about some classes.

Additionally, we utilized several callback functions during training to optimize the model's performance. We used ModelCheckPoint to save the best weights at each training stage and applied ReduceLROnPlateau to automatically reduce the learning rate if the model's validation loss plateaued. We also used EarlyStopping to prevent overfitting by stopping the training before completing the planned number of epochs and restoring the best weights.

E. Additional experiments

To prove if our approach actually improved performance, during the Stanford dataset experimentation, we tested four (4) different models:

- FerNet+EfficientNetB2
- EfficientNetB2 alone
- Only LKA as part of the latent block
- Only CBAM as part of the latent block

Given the obtained results during our previous experiment and as the Tsinghua dataset is significantly larger, we decided to pursue the following model testing:

- FerNet+EfficientNetB2
- EfficientNetB2 alone

V. Test on real-life images

To test our proposed model on real-life data, we tested it using relatives' and friends' images of their dogs. Its outcomes resulted in being very decent because there was too much noise in some of the used images. To scale the testing process, we created a web application so different users could try the model on their own and send feedback on how good or how bad the application predicted the inputted image.

The application is developed using Streamlit (a framework for developing data science applications). When we run the application for the first time, the program downloads the saved model from a server we stored our best Stanford dogs model using an HTTP request, loads the model, and asks users to upload an image. After the user uploads the image, the program tries to predict the dog breed in the image. If the virtual machine has already been used, the program will not download the model again and will only ask the user for an image.

An example of the developed app using one image provided by a friend can be found in fig. 7.

Fig. 7: Dog Breed Predictor App using FerNet

VI. Results

As previously mentioned, our model outperforms all the permutations tested in both benchmarking datasets. One of the biggest concerns about its development was how efficient the resulting model could be. EfficientNetB2 has a total of 7,937,642 for the Stanford Dataset (120 classes), while our approach has 8,053,429 parameters. These results mean that we're only increasing 115,787 in the number of parameters (1.46%); however, on average, during the training stage, both models take a similar time to train. This is because of the encoding section of our module, which reduces the required computation.

Regarding accuracy, for the Stanford Dataset, our model reaches an 89.504% validation accuracy after fine-tuning, while EfficientNetB2 alone reaches 89.140 after fine-tuning. On the other hand, after fine-tuning using the Tsinghua data, our model reaches a validation accuracy of 82.99%, while EfficientNetB2 reaches 82.02%. Something to consider is that EfficientNetB2 has been trained using Imagenet [16]. Therefore, there is a high chance that the EfficientNetB2 on its own posses data leakage and thus performs that well; however, even with our few layers, FerNet improved the performance by training only a relatively reduced number of parameters (without falling in overfitting).

In table II, we present the train and validation accuracy at the epoch where each model achieves its best validation performance.

TABLE II: Models' results

Model	Dataset	Train Accuracy	Val Accuracy
FerNet	Stanford Dogs	94.36%	89.50%
Base Model	Stanford Dogs	97.44%	89.14%
Enc. w/ CBAM	Stanford Dogs	93.46%	89.07%
Enc. w/ LKA	Stanford Dogs	93.19%	88.82%
FerNet	Tsinghua Dogs	92.42%	82.99%
Base Model	Tsinghua Dogs	84.02%	82.02%

VII. Future Work

FerNet has much room for improvement, and much testing is left to do. The approach developed for this project is the peak of the iceberg regarding this topic, and as it was an experimentation-based project, further development could be implementing other modules of attention following the hybrid approach that characterizes EfficientNet. Furthermore, another interesting experiment could be to apply our parallel visual attention encoder to other innovative architectures and study their performance in the same environment. Soon we aim to test the following models as the base model:

- Other versions of EfficientNet [12],
- ResNet [17],
- RegNetY [18].

Another area for potential improvement is to test our models using other datasets, not necessarily related to dogs. Some interesting datasets to test are Food101 [19], DeepWeeds [20], and CalTech UCSD Birds 200 [21]

VIII. Conclusion

In this paper, we presented FerNet, a new module that combines visual attention techniques to improve models' performance. FerNet introduces a parallel processing approach that passes the input through two branches: the base model and the proposed module. This module encodes the output to reduce its dimensionality and then simultaneously processes it across two attention submodules. Then the module combines their results and merges them with the base model's output for further computation. We tested our proposed architecture using the Stanford Dogs and Tsinghua Dogs datasets, achieving improved performance with respect to using the base model alone.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] L. Jiao and J. Zhao, "A survey on the new generation of deep learning in image processing," *IEEE Access*, vol. 7, pp. 172 231–172 263, 2019.

[3] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.

[4] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," *arXiv preprint arXiv:1509.00685*, 2015.

[5] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2017.

[6] S. Woo, J. Park, J.-Y. Lee, and I.-S. Kweon, "Cbam: Convolutional block attention module," in *European Conference on Computer Vision*, 2018.

[7] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu, "Visual attention network," *arXiv preprint arXiv:2202.09741*, 2022.

[8] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[9] M.-H. Guo, Z.-N. Liu, T.-J. Mu, and S.-M. Hu, "Beyond self-attention: External attention using two linear layers for visual tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[10] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei, "Novel dataset for fine-grained image categorization," in *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[11] D.-N. Zou, S.-H. Zhang, T.-J. Mu, and M. Zhang, "A new dataset of dog breed images and a benchmark for fine-grained classification," *Computational Visual Media*, 2020. [Online]. Available: https://doi.org/10.1007/s41095-020-0184-6

[12] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[13] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[14] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," 2020.

[19] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – mining discriminative components with random forests," in *European Conference on Computer Vision*, 2014.

[20] A. Olsen, D. A. Konovalov, B. Philippa, P. Ridd, J. C. Wood, J. Johns, W. Banks, B. Girgenti, O. Kenny, J. Whinney, B. Calvert, M. Rahimi Azghadi, and R. D. White, "DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning," *Scientific Reports*, vol. 9, no. 2058, 2 2019. [Online]. Available: https://doi.org/10.1038/s41598-018-38343-3

[21] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," California Institute of Technology, Tech. Rep. CNS-TR-2010-001, 2010.