Image Understanding by Captioning with Differentiable Architecture Search

Zeting Luan
Department of Electrical and Computer Engineering
UC San Diego
zluan@ucsd.edu

Ziyan Zhu
Department of Mathematics
UC San Diego
ziz276@ucsd.edu

Abstract

Image captioning is a crucial task of generating a description that captures the main contents of an image. With the advancement of Deep Learning Techniques and a larger amount of data, image captioning is achievable by utilizing different Deep Learning approaches. Hence, image captioning is widely used in areas such as aiding the visually impaired or helping social media with better content management. In image captioning, models have encoder-decoder architecture, where the encoders take the input images, produce embeddings, and feed them into the decoders to generate textual descriptions. Manually designing the most suitable image captioning encoder-decoder architectures is a difficult challenge due to the complexity of recognizing the critical objects of the input images and their relationships to generate caption descriptions. To address this problem, we propose a novel three-level optimization method that automatically employs a differentiable architecture search strategy for our image captioning model to search for the most suitable architecture. Inspired by learning from mistakes scheme, we can validate the performance of the trained neural network by evaluating the performance of a student model on validation dataset to further improve the performance of our framework. (We used three grace days for this report)

1 Introduction

Over the years, the most popular image captioning models involve feeding convolutional image features to a recurrent network to generate natural language. Some of the best approaches rely on attention mechanism [1] while Nguyen and Tripathi *et al.* [2] integrated long-short-term-memory network (LSTM) along with the attention mechanism for language decoding. While manually-designed models achieve significant results, it is a difficult and task because of the complexity of recognizing the objects. Neural architecture search (NAS) provides an automatic way of searching for the best models. Several state-of-art NAS architectures involve gradient-based optimization [3][4][5], as opposed to inefficient black-box search. To strengthen the effectiveness of differentiable architecture search strategy for image captioning model, we integrate Learning From Mistakes (LFM) framework [6] for it can be used in combination with any differentiable NAS method to further improve these methods.

2 Related Works

2.1 Neural Architecture Search (NAS)

Neural architecture search (NAS) is a systematic and automatic way of identifying the cost-efficient and highly-performing architectures of neural network models. Recent years, NAS has been adopted to reduce the reliance on human experts by automating the design of neural network topology as learning the parameters of machine models. Prior works on NAS attempted to depict the configurations of neural network operations to construct an optimal complete architecture [7][8] [9][10]. Among those methods, Zoph *et al.* [7] used reinforcement learning (RL) based approach, which trained a recurrent neural network as the controller to output the model descriptions of neural network architectures and evaluated the accuracy of the generated architectures on validation set. Real *et al.* [11] adopted the tournament selection evolutionary algorithms (EA) to discover high-performance network architectures. Despite these RL-based or EA-based methods achieved state-of-the-art performance compared with handcrafted architectures, they were computationally very expensive due to the fact that the architecture search problem is formulated as a black-box discrete optimization problem, which requires numerous number of architecture evaluations. To reduce computational cost, Liu *et al.* [3] first initiated a differentiable architecture search framework, DARTS, which used a continue relaxation on search domain so that architecture parameters and weights can be trained jointly via gradient-based optimization methods. Chen *et al.* [4] further proposed an efficient differentiable search algorithm, called P-DARTS, which progressively increased the depth of the model during training process. Xu *et al.* [5] presented PC-DARTS which efficiently samples substructure from a super-network to alleviate the large memory consumption of DARTS.

2.2 Learning From Mistakes (LFM)

Learning From Mistakes (LFM) is a novel machine learning framework for NAS based on human learning techniques. In LFM, a learner improves its learning ability by revising its own mistakes. LFM was formulated by Garg and Zhang *et al.* [6] as a three-stage optimization problem: 1) learner learns; 2) learner re-learns focusing on the mistakes, and; 3) learner validates its learning [6]. LFM is highly integrated with NAS, which enhances the performance of NAS even further.

2.3 Image Captioning (IC)

Image captioning (IC) is the process of generating textual description of an image based on the objects and actions in the image, i.e., to capture the image relationship between objects, and generate natural language description. It uses both Natural Language Processing (NLP) and Computer Vision (CV) to generate the captions. Most recent image captioning models are attention-based deep-learning models [12] [13]. Most of them encode the image using a CNN, and build and attention-based Recurrent Neural Network (RNN) [14]. Wu *et al.* [15] proposed a novel approach to improve VQA performance by jointly generating captions that are targeted to help answer a specific visual question.

Notation	Meaning
E	Encoder weights
A	Encoder architecture
F	Decoder weights
W	Network weights of image captioning model
D^{tr}	Training dataset for image captioning
D^{val}	Validation dataset for image captioning
U	Unlabeled image dataset

Table 1: Notations

3 Methods

In this section, we propose a three-stage learning from mistake (LFM) framework that can be trained end-to-end, and an efficient optimization algorithm for solving the problem.

3.1 Three-Stage Learning Framework

In our framework, there are three learning stages. The overall scheme contains an encoder-decoder neural network to generate caption descriptions, and an student image captioning model to validates its performances. We also employed a differentiable architecture search strategy for searching the optimal architecture automatically.

Stage I. In stage I, an encoder-decoder network model is trained to create image captions. The encoder takes an image as input and produces an embedding, which is fed into the decoder to decode a textual description. In the first stage, we use the image captioning dataset D^{tr} to train this network by solving the following problem:

$$E^{*}(A), F^{*}(A) = \min_{E,F} \ L(E, A, F, D^{tr}) \tag{1}$$

where E, F denote the network weights of the encoder and the decoder correspondingly, and A denotes the architecture of the encoder. The architecture A is not updated at this stage and is only used to compute the training loss .

Stage II. In stage II, the trained encoder-decoder neural network acts as a teacher model to generate a pseudo image captioning dataset from some unlabeled images, which will be used to train a student image captioning model.

$$W^{*}(E^{*}(A), F^{*}(A)) = \min_{W} \ L(W, U, E^{*}(A), F^{*}(A)) \tag{2}$$

where U denotes some unlabeled images, which will be used to generate pseudo dataset with trained model from stage 1, and W is the weights of image captioning model.

Stage III. In stage III, the architecture will be learned by minimizing the loss of the student image captioning model on validation set.

Putting the above pieces together, the optimization problem can be summarized as below:

$$\begin{aligned} \min_{A} \ & L(W^{*}(E^{*}(A), F^{*}(A)), D^{val}) \\ \text{s.t.} \ & W^{*}(E^{*}(A), F^{*}(A)) = \min_{W} \ L(W, U, E^{*}(A), F^{*}(A)) \\ & E^{*}(A), F^{*}(A) = \min_{E,F} \ L(E, F, A, D^{tr}) \end{aligned} \tag{3}$$

The second constraint and the first constraint in (3) come from the optimization problem in stage I and stage II correspondingly. The objective function represents the optimization problem in stage III.

The process flow is demonstrated in Figure 3.1. In this end-to-end training scheme, E and F will be learned from the training set and generate a pseudo dataset. After the weights W are trained from this pseudo dataset, we evaluate its performances on validation dataset and update the architecture since since the gradient of A depends on W. An update to A will cause the change of W as well. Along the chain $W \rightarrow A \rightarrow (E, F)$, the network weights of the encoder and the decoder are indirectly influenced by W.

Similar to DARTS [3], we learn the architecture A in a differentiable way by optimizing the set of architecture weights $A = \{a\}$. The search space consists of various building blocks, and a weight is assigned to the output of each block representing how important the block is. During inference process, block associated with the largest weight a are kept to constitute the final architecture.

3.2 Optimization Algorithm and Approximate Gradient

An optimization algorithm to solve problem (3) is described in this section, and the overall algorithm is summarized in Algorithm 1. Inspired by Liu, Simonyan and Yang [3], instead of solving the inner

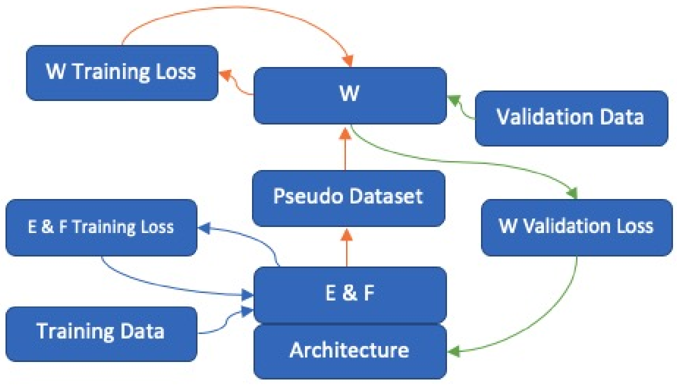

Figure 1: The Overall Process Flow. The blue arrows indicate stage I processes, orange arrows indicate stage II processes, and green arrows indicate stage III processes.

optimization problems in (3) completely, we approximate $E^*(A)$, $F^*(A)$ and $W^*(A)$ by one-step gradient descent to reduce the computational complexity.

Stage I. In stage I, we approximate $E^*(A)$ and $F^*(A)$ using one-step gradient descent for the loss on the training set $L(E, F, A, D^{tr})$ keeping A constant as follows:

$$\begin{aligned} E^*(A) &\approx E' = E - \eta_e \nabla_E L(E, F, A, D^{tr}) \\ F^*(A) &\approx F' = F - \eta_f \nabla_F L(E, F, A, D^{tr}) \end{aligned} \tag{4}$$

where E and F denote the current weights maintained by the algorithm, and η_E and η_F are the learning rates for a step of inner optimization.

Stage II. In stage II, a pseudo dataset is generated by encoder-decoder network with weights E' and F' from (4). Then the weights for student image captioning model are approximated as:

$$W^*(E^*(A), F^*(A)) \approx W' = W - \eta_w \nabla_W L(W, U, E', F') \tag{5}$$

where W denotes the current weights, and η_W is the learning rates.

Stage III. Substituting these approximations E', F' and W' into the objective function in (3), we can compute the update equation for the architecture A as

$$A \leftarrow A - \eta_a \nabla_A L(W', D^{val}) \tag{6}$$

where the gradient of objective function with respect to A can be obtained by chain rule as:

$$\begin{aligned} \nabla_A L(W', D^{val}) =& \eta_e \eta_w \nabla^2_{A,E} L(E, A, F, D^{tr}) \nabla^2_{E',W} L(W, U, E', F') \nabla_{W'} L(W', D^{val}) \\ &+ \eta_f \eta_w \nabla^2_{A,F} L(E, A, F, D^{tr}) \nabla^2_{F',W} L(W, U, E', F') \nabla_{W'} L(W', D^{val}) \end{aligned} \tag{7}$$

Note that if E, F and W are local minimizers for the inner optimization problems, i.e. $\nabla_E L(E, A, F, D^{tr}) = 0$, $\nabla_F L(E, A, F, D^{tr}) = 0$ and $\nabla_W L(W, U, E', F') = 0$, then equation (7) will reduce to $\nabla_A L(W, D^{val})$. With properly chosen η_e, η_f and η_w, we expect that this algorithm converges to a fixed point solution.

To avoid the expensive matrix-vector product, we can use finite difference approximation. Recall the formula of finite difference method for approximating the directional derivative at current iterate x along the direction h:

$$\nabla f(x)^T h \approx \frac{f(x + \epsilon h) - f(x - \epsilon h)}{\epsilon} \tag{8}$$

where $\epsilon \in \mathbb{R}$ is a small scale value. Let ϵ_w, ϵ_e and ϵ_f be some small scales values. Applying the formula (8) to the matrix-vector products in (7) twice, we obtain the following approximations.

$$\nabla^2_{E',W} L(W, U, E', F') \nabla_{W'} L(W', D^{val}) \approx \frac{\nabla_{E'} L(W^+, U, E', F') - \nabla_{E'} L(W^-, U, E', F')}{2\epsilon_w} \tag{9}$$

where $W^{\pm} = W' \pm \epsilon_w \nabla_{W'} L(W', D^{val})$. And

$$\nabla^2_{A,E} L(E, A, F, D^{tr}) \nabla_{E'} L(W^{\pm}, U, E', F') \approx \frac{\nabla_A L(E^+, A, F, D^{tr}) - \nabla_A L(E^-, A, F, D^{tr})}{2\epsilon_e} \tag{10}$$

where $E^{\pm} = E' \pm \epsilon_e \nabla_{E'} L(W^{\pm}, U, E', F')$. Similarly,

$$\nabla^2_{F',W} L(W, U, E', F') \nabla_{W'} L(W', D^{val}) \approx \frac{\nabla_{F'} L(W^+, U, E', F') - \nabla_{F'} L(W^-, U, E', F')}{2\epsilon_w} \tag{11}$$

where $W^{\pm} = W' \pm \epsilon_w \nabla_{W'} L(W', D^{val})$. And

$$\nabla^2_{A,F} L(E, A, F, D^{tr}) \nabla_{F'} L(W^{\pm}, U, E', F') \approx \frac{\nabla_A L(E, A, F^+, D^{tr}) - \nabla_A L(E, A, F^-, D^{tr})}{2\epsilon_f} \tag{12}$$

where $F^{\pm} = F' \pm \epsilon_f \nabla_{F'} L(W^{\pm}, U, E', F')$. Evaluating finite difference only needs two forward passes for weights E, F and W correspondingly, which could reduce computational complexity significantly from $\mathcal{O}\left((|E| + |F|) \cdot |W| \cdot |A|\right)$ to $\mathcal{O}(|E| + |F| + |W| + |A|)$.

Algorithm 1 Optimization algorithm algorithm for image understanding by textual descriptions

```
1: while not converged do
2:     Update encoder weights E and decoder weights F using Eq.(4)
3:     Update image captioning model weights W using Eq.(5)
4:     Update encoder architecture A using Eq.(6)
5: end while
```

4 Experiments

In this experiments, our proposed three-stage LFM framework will be applied to perform neural architecture search for image captioning task. To evaluate our framework, we compared it with human designed image captioning network proposed by Xu *et al.* [16] (baseline model) and a encoder-decoder network learnt by DARTS.

4.1 Dataset and Evaluation

Due to limited computational recourse and time limit, the experiments are performed on the popular image captioning dataset, Flickr8k [17]. The Flickr8k dataset contains a total of 8092 images in JPEG format with various shapes and sizes, and there are five reference captions per image. We use publicly available splits used in previous work [18]. where the dataset is split into three sets, 6000 images in training set, 1000 images in validation set and 1000 images in test set.

The frequently used BiLingual Evaluation Understudy (BLEU) [19] is used as the evaluation metric in this report, which is the standard in the caption generation literature. BLEU evaluates a generated caption against reference captions and its value lies between 0 and 1 (higher is better). For each generated caption, all five captions available for that image are used as the reference captions. A smoothing function is applied to avoid getting zero for BLEU score because the precision for the order of n-grams without overlap is zero, which leads to inaccurate result. In this report, we consider BLEU-4 score, i.e. 4-grams.

4.2 Experimental Settings

In our experiment, the three-stage LFM framework is applied to DARTS for automatically searching for the optimal architecture of encoder network while the decoder network architecture is fixed. We used the same search spaces as in DARTS: separable convolutions and dilated separable convolutions with sizes of 3×3 and 5×5, max pooling with the size of 3×3, average pooling with the size of 3×3, identity, and zero. In this report, we focused on lightweight and cost-efficient architectures for mobile usage.

4.2.1 Architecture Search

During the architecture searching process, the architecture of a small encoder network of 8 cells, where each cell is made of 7 nodes, will be learned by training for 50 epochs with early stopping. Similar to DARTS, there are two types of cells, normal cell and reduction cell. Reduction cells are located at the $\frac{1}{3}$ and $\frac{2}{3}$ of the total depth of the network, in which all the operations adjacent to the input nodes are of stride two. A global adaptive average pooling layer is applied at the end of encoder network to resize the encoding to a fixed size. We used Adam optimizer for learning the architecture weights with initial learning rate 5×10^{-3}. In order to fit into a single GPU, the batch size is selected to be 32 for both the training and validation sets and the image input size is set to be 256×256. To ensure our model size is comparable with the baseline model, the initial number of channels of encoder network is chosen to be 32. For the decoder network, a LSTM Network with attention [16], which is a popular choice for image captioning model, is used in our experiment. The hidden and cell state of the LSTM is initialized with the encoded image from encoder network. SDG with momentum is used for training encoder and decoder with an initial learning rate of 2.5×10^{-2} and 4×10^{-4} respectively. For student models, the encoder network is based on MobileNet V2 architecture while the decoder network has the same architecture as teacher decoder. SDG with momentum is used for training student model with an initial learning rate of 1×10^{-4}. The model was trained for about two days on a single GPU.

4.2.2 Architecture Evaluation

During architecture evaluation process, the weights of model with selected topology will be initialized randomly and trained from scratch and the architecture search is discarded. Then, the model is evaluated on the testing set (note we never used testing set for training or searching processes). 25 layer copies of optimally searched cells are stacked into a large network. We trained the model for 75 epochs with early stopping on a single GPU. All other hyper-parameters settings, such as optimizer, learning rate and batch size, are the same as these numbers in the architecture searching process.

During the inference process, we performed Beam Search [20] for generating the texture image captions. In particular, instead of greedily choosing the word with the highest score and using it to predict the next word, we select the sequence that has the highest overall score from a set of candidate sequences. In our experiment, we considered top 3 candidates from each output.

4.2.3 Baseline Model

The baseline model we choose is manually designed image captioning network proposed by Xu *et al.* [16]. The model generates sequences will use an encoder network to encode the input into a fixed form and a decoder network to decode it, word by word, into a sequence. The encoder network is a convolutional neural networks that encodes the input image into an embedding representations, while the decoder network is a LSTM model that takes the encoded embedding as input and generate a caption word by word. Moreover, Xu *et al.* applied attention mechanism which allows the model to focus on the pixels of the image most relevant to the word that it is going to generate next. In our experiment, the decoder network architecture is fixed and we perform the experiment of encoder network with different backbones, ResNet18 and MobileNetV2. The other details are are kept same as the experiments described in earlier sections.

4.3 Results

In this section, we will show some image captions generated by the automatically searched encoder-decoder network and visualize our models by plotting their cell architectures. The BLEU-4 scores and parameter sizes of these models will be compared and discussed.

4.3.1 Some Generated Image Captions

In the Figure 2, four images from the testing set of Flickr8k are selected and fed into our encoder-decoder network trained by LMF-DARTS. From Figure 2, we can observe that the trained image caption model is surprisingly good and is able to generate reasonable captions for these four images. Specifically, in the most left image, our model could detect the wall is decorated and the person is a female.

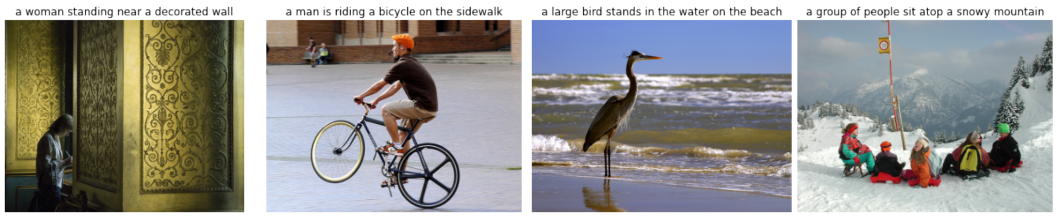
a woman standing near a decorated wall
a man is riding a bicycle on the sidewalk
a large bird stands in the water on the beach
a group of people sit atop a snowy mountain

Figure 2: Some image captions generated by searched model by LFM-DARTS

4.3.2 Architecture Visualization

In this section, we visualize the searched model by plotting the obtained genotype. The architectures of normal cell and the reduction cell learned by DARTS at 40-th epoch are shown in Figure 3. The architectures of normal cell and the reduction cell learned by LFM-DARTS at 35-th epoch are shown in Figure 4.

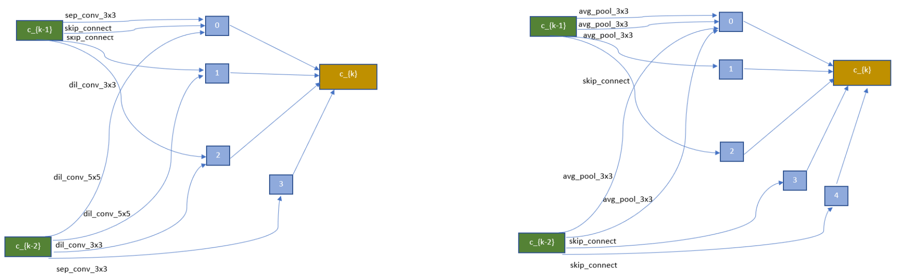

Figure 3: Genotypes of Normal cell (left) and Reduction cell (right) learned by DARTS

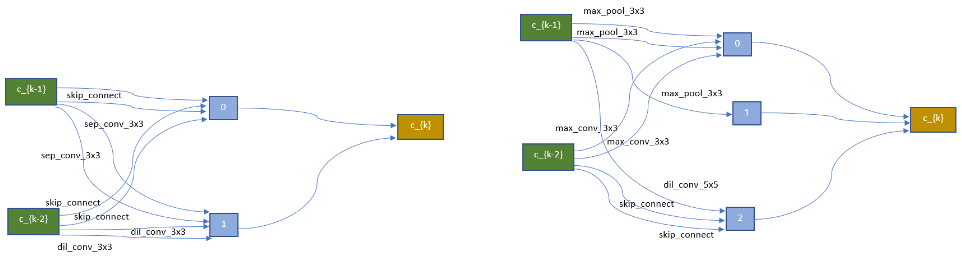

Figure 4: Genotypes of Normal cell (left) and Reduction cell (right) learned by LFM-DARTS

4.3.3 Analysis of results

The results of BLEU-4 score of different models evaluated on Flickr8k dataset are shown in Table 2. From top to bottom, on the first two blocks are the models that we implemented, manually designed baseline models and differentiable NAS methods. The third blocks are some manually designed state-of-the-art models on Flickr8k dataset.

Method	Param Size	BLEU-4
Baseline-ResNet18	44.8 MB	0.2219
Baseline-MobileNetV2	9.14 MB	0.1976
LFM-DARTS (ours)	7.48 MB	0.1724
DARTS	5.74 MB	0.13
Karpathy and Li (NeuralTalk)[21]	-	0.16
Chen and Zintick (Mind's Eye)[22]	-	0.16
Google (NIC)[23]	-	0.16
Mao *et al.* (m-Rnn)[24]	-	0.17
Wu *et al.* (VggNet+LSTM)[25]	-	0.16
Jia *et al.* (GLSTM)[26]	-	0.21

Table 2: BLEU-4 score on on the test set of Flickr8k.

We compared the results of our model with different state-of-the-art models. With only 7.48 MB model parameter size, our LFM-DARTS model reaches about 0.1724 in BLEU-4 score, which is the same as the m-Rnn model by Mao *et al.* [24] and slightly higher than some of the other models listed above. The BLEU-4 score of our model did not exceed the score in the baseline models. We speculate the reasons as follows: 1) Due to the limitation of time and computational resource, we did not run on a bigger dataset like MSCOCO 2015 [27], and DARTS could have a better performance on bigger dataset; 2) The hyper-parameters could be tuned more precisely for better performance; 3) The model was not fully converged as the training epochs under 100; 4) Our model has a simpler architecture and smaller model size than baseline models. On the other hand, although the BLEU-4 score of DARTS is only at 0.13, the LFM-DARTS improves the score to 0.1724. Therefore the three-stages optimization framework indeed improves the performance of DARTS by a considerable margin.

5 Conclusions

In this report, we presented a three-stage learning from mistake framework and applied it to differential Neural Architecture Search method. The learner model acts as a teacher to generate pesudo caption dataset and improves its learning ability by teaching a student image caption model. Moreover, we proposed an trilevel optimization framework and an efficient algorithm to solve it. Furthermore, we performed extensive numerical experiments to compare our models with state-of-the-art models and showed our model has a comparable performance with smaller model size.

References

[1] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning, 2017.

[2] Kien Nguyen, Subarna Tripathi, Bang Du, Tanaya Guha, and Truong Q. Nguyen. In defense of scene graphs for image captioning, 2021.

[3] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[4] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1294–1303, 2019.

[5] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient architecture search. *arXiv preprint arXiv:1907.05737*, 2019.

[6] Bhanu Garg, Li Zhang, Pradyumna Sridhara, Ramtin Hosseini, Eric Xing, and Pengtao Xie. Learning from mistakes – a framework for neural architecture search, 2022.

[7] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[8] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[9] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International Conference on Machine Learning*, pages 4095–4104. PMLR, 2018.

[10] Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[11] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[12] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions, 2015.

[13] Xihui Liu, Hongsheng Li, Jing Shao, Dapeng Chen, and Xiaogang Wang. Show, tell and discriminate: Image captioning by self-retrieval with partially labeled data, 2018.

[14] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[15] Jialin Wu, Zeyuan Hu, and Raymond J. Mooney. Generating question relevant captions to aid visual question answering, 2020.

[16] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[17] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.

[18] Kingma Da. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[20] Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. *arXiv preprint arXiv:1702.01806*, 2017.

[21] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *CoRR*, abs/1412.2306, 2014.

[22] Xinlei Chen and C. Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2422–2431, 2015.

[23] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.

[24] Junhua Mao, Xu Wei, Yi Yang, Wang Jiang, and Zhiheng Huang Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*, abs/1412.6632, 2015.

[25] Qi Wu, Chunhua Shen, Anton van den Hengel, Peng Wang, and Anthony R. Dick. Image captioning and visual question answering based on attributes and their related external knowledge. *CoRR*, abs/1603.02814, 2016.

[26] Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. Guiding long-short term memory for image caption generation. *CoRR*, abs/1509.04942, 2015.

[27] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server, 2015.