Molecular Translation: Interpreting Organic Compound Images with Image Captioning

Boyuan Liu

Georgia Institute of Technology
bryanliu@gatech.edu

Abstract

Organic compounds are prevalent in the fields of chemistry and biology. They are often represented in diagrams of skeletal formulas to give comprehensive overviews of their structures. Searching for organic compounds in the diagrams in literature is not an easy task and often requires inefficient manual work. In this project, we built a versatile image captioning pipeline that can adapt to different tasks by switching the models in both the encoder and decoder. The EfficientNet B2 encoder + Transformer decoder structure had the best performance among the different setups tested in the experiments and achieved 96.9% accuracy in predicted sequences.

1. Introduction

The advances in image captioning in the past few years have made many previously unachievable tasks possible. Researchers went from describing dogs in images to assisting visually impaired people [1]. In the field of organic chemistry, chemists have been using the skeletal formula, a structural notation of chemical compounds, for centuries. These image representations provide intuitive overviews of the chemicals, but they are very challenging to search for in the literature. Finding the exact organic compound often requires inefficient manual work since there are estimated to be nearly 24 million organic compounds [2]. Researchers often find themselves digging through piles of documentation, searching for information related to an organic compound. Automated recognition of optical chemical structures via image captioning would help researchers speed up this process significantly. Previous research have used methods such as the rule-based OSRA [3], an approach based on OpenNMT [4], and a feature extraction-assembly approach called OCR [5]. These methods have various limitations, such as accuracy, the complexity of the organic compounds, or image quality requirements.

In this paper, we implemented the widely adopted encoder-decoder structure in image captioning and experimented with various state-of-the-art image models to find the best architecture for this task that balances efficiency and performance. The dataset used in this project contains approximately four million images of organic compounds synthetically generated by Bristol-Myers Squibb (BMS) [6]. These images vary in rotation angles, resolutions, and noise levels. This project aims to build an efficient image captioning pipeline that can process batches of organic compound images and accurately output their International Chemical Identifier (InChI) [7] text representation.

2. Related Work

EfficientNet: Scaling up neural networks by their width and depth is a common way to improve their performance on large datasets. Scaling network depth is the most common way used by many ConvNets [8] [9][10], while scaling network width is commonly used for small-size models [11][12]. Scaling up input image resolutions can also improve accuracy since it can potentially help neural networks capture more fine-grained patterns [13]. However, the scaled-up models could be over-parameterized, which leads to model inefficiency. The EfficientNet by Quoc et al. proposed a principled method to scale up ConvNets to achieve better accuracy and efficiency [14]. It used a compound scaling method to uniformly scale up the

model's width, depth, and resolution. The EfficientNet-B7 model achieved state-of-the-art 84.3% accuracy on ImageNet while being 8.4x smaller and 6.1x faster on inference than the ConvNet [14].

Transformer: The Transformer is the first transduction model relying entirely on self-attention to compute its outputs, which makes it significantly faster to train than architectures based on recurrent or convolutional layers [15]. The attention function maps a query and a set of key-value pairs to an output. The "Scaled Dot-Product Attention" used in Transformers computes the dot product of the query with all keys, scaled by the input dimension, and applies a softmax function to obtain the weights for the values [15]. These weighted values can be viewed as values with added context information. The encoder and decoder modules often have multiple self-attention blocks stacked together for better performance. The Transformer model outperformed the previous best models in WMT 2014 English-to-French and English-to-German with a fraction of the training cost [15].

Image Captioning: Image captioning utilizes a combination of computer vision and natural language processing techniques to predict captions based on visual inputs. The multimodal log-bilinear model introduced by Kiros et al. [16] was the first attempt to use neural networks for image caption generation. Mao et al. first used a recurrent neural network instead of a feed-forward model [17]. The widely cited *Show, Attend and Tell* paper [18] introduced a CNN encoder + LSTM with attention mechanism decoder architecture. The CNN encoder extract feature information from the images, and the LSTM decoder generates text sequences based on the extracted features. As a result, the model was able to learn alignments that corresponded very strongly with human intuition and achieved state-of-the-art performance on the Flickr8k, Flickr30k, and MS COCO benchmark datasets [18].

3. The Bristol-Myers Squibb Dataset

The dataset used in this project was synthetically generated by Bristol-Myers Squibb. It contains nearly four million images of organic compounds, including 2,344,186 training plus validation examples and 1,616,107 testing examples. The images in the dataset have various resolutions, noise levels, and rotation angles. The labels are the InChI text representation of the corresponding organic compound. Figure 1 shows a typical training example in the dataset. The lower left region of the image has some corruption. The image contains letters N, H, and O, representing nitrogen, hydrogen, and oxygen atoms, '—' and '==' representing the single and double bonds between atoms, and '⌬' representing benzene rings. The organic compound shown in Figure 1 has an InChI label length of 197. The training examples in this dataset can have label lengths up to 277.

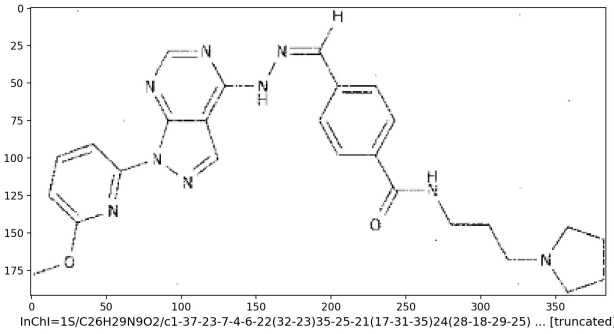


Figure 1 - A training example from the dataset. Letters N, H, and O, represent nitrogen, hydrogen, and oxygen atoms, '—' and '==' represent the single and double bonds between atoms, and the '⌬' symbol represents a benzene ring. Its InChI label length is 197.

4. E2E Image Captioning Architecture

Molecular translation is an image captioning task at its core. Image captioning is a versatile application as it allows us to automate the task of generating text representations for any image. In this project, we built an end-to-end image captioning pipeline. This E2E pipeline used a preprocessing-encoder-decoder-output architecture that is similar to the architecture used in the widely cited *Show, Attend and Tell* paper [ ]. To make the image captioning pipeline more versatile, we designed the pipeline in a way that allows easy

switches among models. Instead of using a fixed model for the encoder and another fixed model for the decoder, the image captioning pipeline can choose among EfficientNet, MobileNet, ResNet, Vision Transformer, or Transformer in Transformer for the encoder and LSTM or Transformer for the decoder. This design provides flexibility and makes it easier to search for the best models for a given task.

4.1 CNN/Vision Transformer Encoder

The E2E pipeline has an encoder block that can adopt a variety of CNN or Vision Transformer models. The encoder takes preprocessed organic compound images as inputs and performs feature extraction to generate embedded feature vectors that will be passed as input to the decoder. Some of the implemented models are EfficientNet, MobileNet, ResNet, Vision Transformer, and Transformer in Transformer. The EfficientNet was the primary model used in the molecular translation task. EfficientNet is a state-of-the-art computer vision architecture. It uses a compound scaling technique to uniformly scales model width, depth, and resolution in a principled way [14] with a compound coefficient φ. The scaling factors are calculated using the equations below, while α, β, γ specify how much extra resources are assigned to network width, depth, and resolution respectively. EfficientNet B0 to B5 were tested and compared to the other vision models in the experiments.

$$\begin{aligned} \text{depth: } & d = \alpha^{\phi} \\ \text{width: } & w = \beta^{\phi} \\ \text{resolution: } & r = \gamma^{\phi} \\ \text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

4.2 Transformer/LSTM Decoder

Similar to the encoder, the decoder can choose between LSTM and Transformer models. For the LSTM model, we leveraged a two-layer LSTM with attention mechanism that closely follows the one used in Zaremba et al [18][19]. The feature vectors from the encoder are fed in as input to the attention and LSTM layers, which generate a predicted text sequence. LSTM layers are defined by the following equations, where, i_t, o_t, f_t, c_t, h_t are the input, output, forget, memory and hidden state respectively. The vector $\hat{\mathbf{z}}$ is the context vector and $\mathbf{E}$ is the embedding matrix [18]. We chose this architecture to take advantage of LSTM's ability to counter the vanishing gradient problem as the sequence length increases. The attention mechanism allows the decoder to attend to all the hidden states from the encoder and focus on certain parts of the input to better understand its context.

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t).$$

The Transformer model further utilizes the attention mechanism. Instead of using a single attention layer before the LSTM layers, the Transformer model can stack several multi-head attention blocks together. The self-attention blocks use the following equation to compute the dot product of the query with all keys, scaled by the input dimension, and apply a softmax function to obtain the weights for the values [15]. This weighted values can be viewed as values with added context information. This ultimately leads to better predicted sequences and faster training.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Figure 2 shows the CNN + Transformer architecture. The CNN model takes preprocessed images and extracts features that are later fed into the Transformer encoder. The outputs from the Transformer encoder are used as keys and values by the decoder blocks to generate sequences.

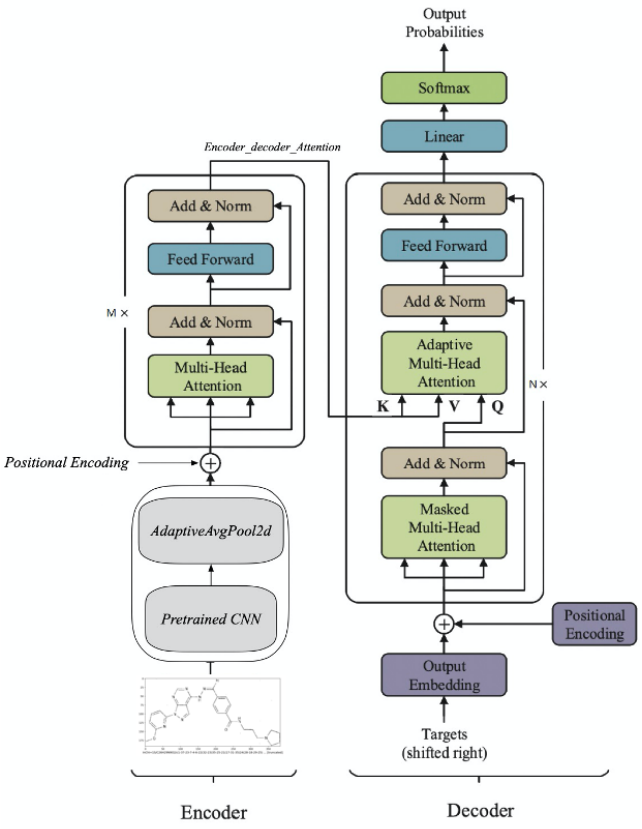


Figure 2 - The CNN + Transformer Architecture [15]. The CNN model takes preprocessed images and extracts features that are later fed into the Transformer encoder. The outputs from the Transformer encoder are used as keys and values by the decoder blocks to generate sequences.

5. Results and Discussion

Several experiments were conducted to find the optimal encoder/decoder setup for the E2E pipeline. Due to the large size of the training set, these experiments used a randomly sampled subset of 10,000 images for training and validation. The models were trained with the full dataset after the optimal setup was determined.

5.1 Transformer v.s. LSTM

We adopted the LSTM with attention architecture introduced by Xu, Kelvin, et al. [18] and implemented a Transformer decoder as an alternative option. To compare the performance of LSTM and Transformer decoders, we used a pre-trained EfficientNet B0 as the encoder and trained both decoders using the same hyperparameters. We used the Levenshtein distance [20] and accuracy as the primary metrics to evaluate the model's performance. The validation accuracy and normalized Levenshtein distance are shown in Figure 3. We used the moving average with a sliding window of 5 epochs to better show the overall trend. The Transformer decoder had higher accuracy and lower Levenshtein distance after 60 epochs of training. The LSTM decoder also appears to converge sooner at 20 epochs, while the Transformer decoder's accuracy was still improving after 60 epochs. The EfficientNet + LSTM setup took 240 seconds per epoch on average. On average, the EfficientNet + Transformer setup took 142 seconds per epoch, which is 40% faster. The Transformer decoder had better performance and faster training time. Therefore, the Transformer is the better decoder option for the molecular translation task.

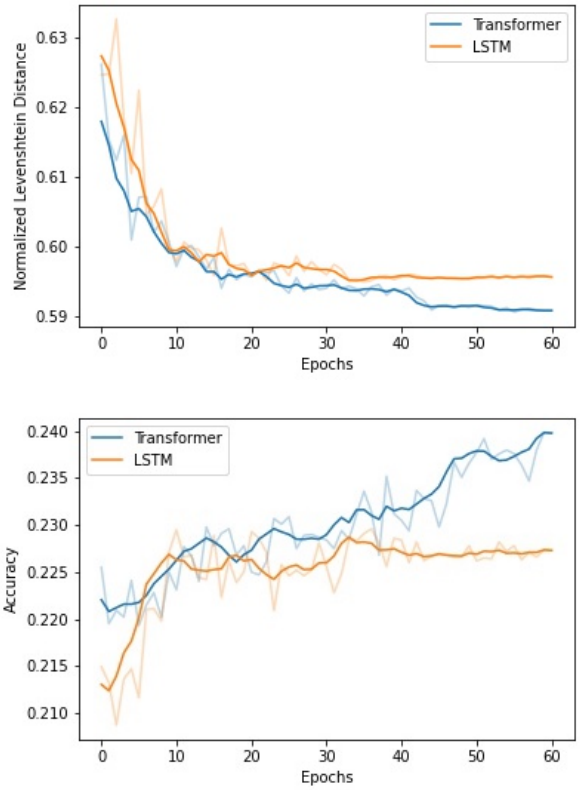
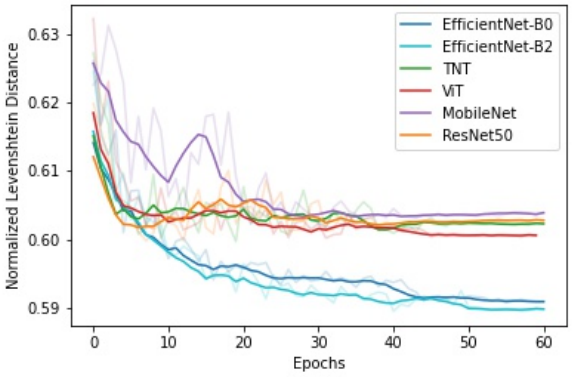


Figure 3 - Validation Accuracy and Normalized Levenshtein Distance of LSTM and Transformer. The Transformer decoder had higher accuracy and lower Levenshtein distance after 60

epochs. The EfficientNet + Transformer setup was also 40% faster than the EfficientNet + LSTM setup.

5.2 Encoder Vision Model Comparison

After choosing the Transformer model as the decoder, we used the same method to compare several state-of-the-art models for the encoder, including EfficientNet, MobileNet, ResNet, Vision Transformer (ViT), and Transformer in Transformer (TNT). These encoder models were paired with the same Transformer decoder and trained using the same hyperparameters. Figure 4 shows the normalized Levenshtein distance and accuracy during validation. We used the moving average with a sliding window of 5 epochs to better show the overall trend. EfficientNet B0 and B2 outperform the other models by a prominent margin in terms of both validation accuracy and Levenshtein distance. EfficientNet B2 had the third-best training time of 178 seconds per epoch on average, closely behind EfficientNet B0 and MobileNet. The details of the models' performance after 60 epochs of training are shown in Table 1. The EfficientNet B2 + Transformer structure was chosen eventually after balancing performance and training time.



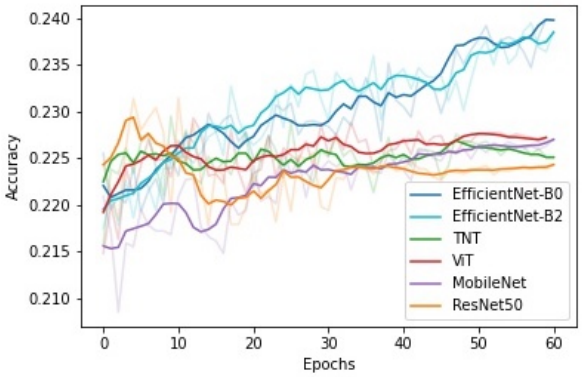


Figure 4 - Validation Accuracy and Normalized Levenshtein Distance of Encoder Models. EfficientNet B2 outperforms the other models by a prominent margin while having the third-best training time, closely behind EfficientNet B0 and MobileNet.

	Normalized Levenshtein	Accuracy	Time (s) / Epoch
EfficientNet B2	0.5898	0.2385	178
EfficientNet B0	0.5909	0.2398	142
ViT	0.6006	0.2272	253
TNT	0.6023	0.2251	261
ResNet50	0.6028	0.2243	180
MobileNet	0.6039	0.2270	159

Table 1 - Encoder models' performance after 60 epochs of training on the 10,000 images subset (ranked by their normalized Levenshtein distances).

5.3 Final Results

Based on the results from the previous experiments, the EfficientNet B2 encoder + Transformer decoder structure has the best overall performance on the 10,000 images subset. After hyperparameter tuning, this setup was trained with the full 4 million images dataset for 45 epochs. The validation results in Figure 5 show that the training accuracy reached 99.7% at the end of the training, while the validation accuracy was lower by a small margin. This indicates that the model experienced a certain level of overfitting. The

model reached high accuracy and low Levenshtein distance after ten epochs and continued improving until it converged at 40 epochs. The model eventually achieved 96.9% accuracy and 1.88 Levenshtein distance in validation and 3.58 Levenshtein distance in testing. It performed better than some of the approaches in previous research, such as OSCR and CSR [5], by a noticeable margin.

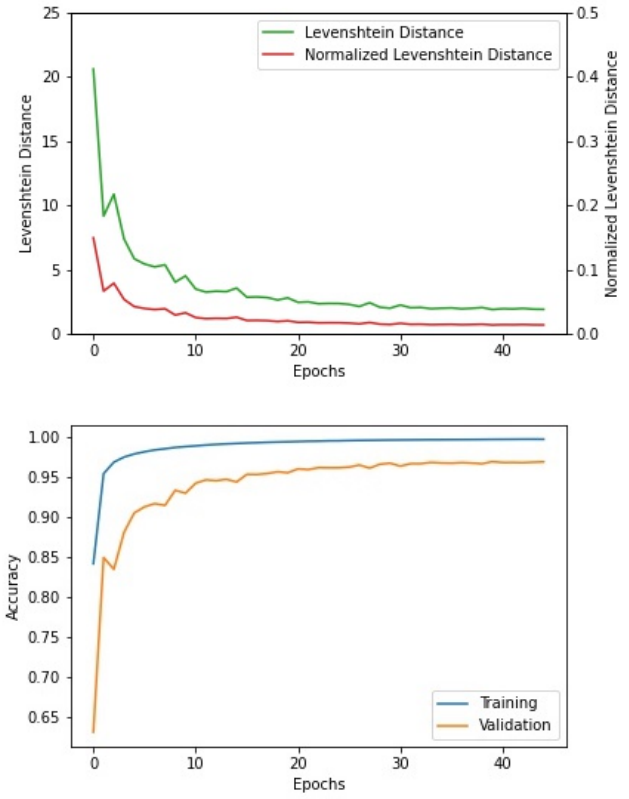


Figure 5 - Accuracy and Levenshtein Distance of the Final Model. The model eventually achieved 96.9% in accuracy and 1.88 in Levenshtein distance after converging at 40 epochs.

6. Conclusion

In this project, we built an end-to-end image captioning pipeline for the molecular translation task. The image captioning pipeline incorporated different models such as EfficientNet, ResNet, Transformer, and LSTM for both the encoder and the decoder. After the experiments, the EfficientNet B2 encoder + Transformer decoder structure proved to be the optimal setup for the molecular translation task. The model eventually achieved 96.9% accuracy and 1.88 Levenshtein distance in predicted sequences. This image captioning pipeline's ability to easily switch among models for both encoder and decoder provides excellent versatility, which can help the pipeline adapt to other image captioning tasks.

References

[1] Makav, Burak, and Volkan Kılıç. "A new image captioning approach for visually impaired people." *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*. IEEE, 2019.

[2] Lipkus, Alan H., et al. "Structural diversity of organic chemistry. A scaffold analysis of the CAS Registry." *The Journal of organic chemistry* 73.12 (2008): 4443-4451.

[3] Filippov, Igor V., and Marc C. Nicklaus. "Optical structure recognition software to recover chemical information: OSRA, an open source solution." (2009): 740-743.

[4] Tabchouri, Sophia. A machine learning approach to molecular structure recognition in chemical literature. Diss. Massachusetts Institute of Technology, 2019.

[5] Bukhari, Syed Saqib, Zaryab Iftikhar, and Andreas Dengel. "Chemical Structure Recognition (CSR) System: Automatic Analysis of 2D Chemical Structures in Document Images." *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019.

[6] BMS-Molecular-Translation. 2021. Available online: https://www.kaggle.com/c/bms-molecular-translation

[7] Heller, Stephen R., et al. "InChI, the IUPAC international chemical identifier." *Journal of cheminformatics* 7.1 (2015): 1-34.

[8] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[9] Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[10] Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
[11] Howard, A. G., et al. "Efficient convolutional neural networks for mobile vision." *arXiv preprint arXiv:1704.04861* (2017).
[12] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
[13] Huang, Yanping, et al. "Gpipe: Efficient training of giant neural networks using pipeline parallelism." *Advances in neural information processing systems* 32 (2019).
[14] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.
[15] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).
[16] Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel. "Multimodal neural language models." *International conference on machine learning*. PMLR, 2014.
[17] Mao, Junhua, et al. "Deep captioning with multimodal recurrent neural networks (m-rnn)." *arXiv preprint arXiv:1412.6632* (2014).
[18] Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." *International conference on machine learning*. PMLR, 2015.
[19] Zaremba, Wojciech, Ilya Sutskever, and Oriol Vinyals. "Recurrent neural network regularization." *arXiv preprint arXiv:1409.2329* (2014).
[20] Yujian, Li, and Liu Bo. "A normalized Levenshtein distance metric." *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007): 1091-1095.