Generating Webpages from Screenshots

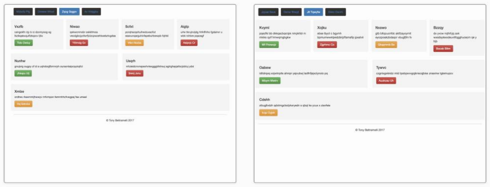
Andrew S. Lee
andrewslee@stanford.edu

Abstract

This project created a PyTorch implementation of an image-captioning model in order to convert screenshots of webpages into code, following pix2code[1]. The system passes images into a ResNet-152-based CNN encoder model, which generates features for a custom decoder RNN model. The project resulted in peak BLEU scores plateauing around `0.92` after a few hundred epochs.

Data

Our data used pix2code's generated screenshots based on a Bootstrap-based DSL vocabulary (18 words). It contains `1,750` pairs of `2400×1380` color images and their associated DSL code. We converted the image dimensions to `224×224` to use with ResNet-152.



Actual

header { btn-inactive, btn-inactive, btn-active, btn-inactive } row { quadruple { small-title, text, btn-green } quadruple { small-title, text, btn-red } quadruple { small-title, text, btn-orange } quadruple { small-title, text, btn-red } } row { double { small-title, text, btn-green } double { small-title, text, btn-red } } row { single { small-title, text, btn-orange } }

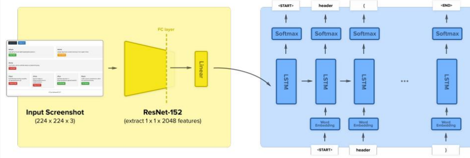
Predicted

header { btn-inactive , btn-inactive , btn-active , btn-inactive } row { quadruple { small-title , text , btn-green } quadruple { small-title , text , btn-red } quadruple { small-title , text , btn-orange } quadruple { small-title , text , btn-red } } row { double { small-title , text , btn-green } double { small-title , text , btn-red } } row { single { small-title , text , btn-orange } }

***Above**: Example of target and predicted web pages (and DSL).*

Models

All of our features are gathered from a pre-trained ResNet-152 (size `1×1×2048` per screenshot) model. While the model was not trained on GUI images[3], it does surprisingly well at extracting backgrounds, edges, colors, and text. This meant it was an easy and appropriate base to build our system on.



Encoder Model

The encoder model is based on a pre-trained ResNet-152 model. We replace the final collection layer in order to collect a feature vector, which we then pass through a linear layer.

Decoder Model

The decoder model takes as inputs 1) the extracted features from the encoder model and 2) their target captions (DSL code put into a word embedding). It uses an LSTM, which we teach a language model based on the inputted features.

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{(t-1)} + b_{hi}) & o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) & c_t &= f_t c_{(t-1)} + i_t g_t \\
g_t &= \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) & h_t &= o_t \tanh(c_t)
\end{aligned}
$$

***Above**: Equations for multi-layer LSTM RNN.*

Results

We are using Bilingual Evaluation Understudy Scores (BLEU) to quantify our results, which is common for image-captioning models[2].

Model	Training (BLEU Score)	Test (BLEU Score)	Train Set (#)	Dev Set Size (#)	Test Set Size (#)
`100 epochs hidden_size=512`	0.95	0.92	1360	170	170
`500 epochs hidden_size=512`	0.99	0.90	1360	170	170
`100 epochs hidden_size=256`	0.85	0.76	1360	170	170
`500 epochs hidden_size=256`	0.93	0.84	1360	170	170

Discussion

The most surprising part of this project's success is how well a pre-trained image model can extract features from graphical interfaces, especially because they're not trained on them. However, we suspect that the pre-trained model is the source of most of the existing error, particularly around color-detection. What makes the system effective at the moment is likely the very simple DSL language. It would be interesting to experiment with a broader vocabulary (2+ orders of magnitude larger) and see if the BLEU scores hold up.

Future

There is definitely room for more exploration — at this point, the system is more of a proof of concept to expand on. We wanted to create an end-to-end model which eliminates the Bootstrap-based DSL and pre-trained CNN, but lacked the time get it working. There is also more room to tweak hyper-parameters and experiment further.

References

[1] Tony Beltramelli. pix2code: Generating code from a graphical user interface screenshot. arXiv preprint arXiv: 1705.07962, 2017.

[2] 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 on Association for Computational Linguistics, ACL '02, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning forimage recognition. CoRR, abs/1512.03385, 2015.