

SANTA CLARA UNIVERSITY

Department of Computer Science and Engineering

I HEREBY RECOMMEND THAT THE THESIS PREPARED
UNDER MY SUPERVISION BY

Federico Madden

ENTITLED

**EXTRACTING CREATIVE
PROCEDURAL KNOWLEDGE FROM
PHOTOSHOP TUTORIALS**

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

**BACHELOR OF SCIENCE
IN
COMPUTER SCIENCE AND ENGINEERING**

Zhigiang Tao

06/09/2021

Thesis Advisor(s)

date

Nam Ling

Nam Ling (Jun 11, 2021 10:56 PDT)

Jun 11, 2021

Department Chair(s) (use separate line for each chair)

date

Extracting Creative Procedural Knowledge from Photoshop Tutorials

By

Federico Madden

SENIOR DESIGN PROJECT REPORT

Submitted to
the Department of Computer Science and Engineering

of

SANTA CLARA UNIVERSITY

in Partial Fulfillment of the Requirements
for the degree of
Bachelor of Science in Computer Science and Engineering

Santa Clara, California

Spring 2021

Abstract

For my senior design project, I created a web app that, given the text of an online Photoshop tutorial, uses a neural network to extract specific Photoshop actions (e.g. File - New or Layers - New Layer) that the tutorial either implicitly or explicitly instructs the reader to take. This neural network consists mainly of a BERT embedding layer followed by an LSTM module, with a linear multi-class, multi-label classification layer on top. The network achieved a macro average F1 score of 15.1% on tutorial sentence classification tasks.

Contents

| | | |
|-----------|--|-----------|
| 1 | Introduction & Problem Statement | 3 |
| 2 | Literature Review | 3 |
| 3 | Project Risks | 3 |
| 4 | Use Cases | 4 |
| 5 | Conceptual Model | 4 |
| 6 | Systems Level Chapter | 5 |
| 6.1 | User Interface | 5 |
| 6.1.1 | Requirements | 5 |
| 6.1.2 | UI Mockup | 6 |
| 6.1.3 | Design Rationale | 7 |
| 6.1.4 | Technologies Used | 7 |
| 6.2 | Neural Network | 7 |
| 6.2.1 | Requirements | 7 |
| 6.2.2 | Design Rationale | 9 |
| 6.2.3 | Technologies Used | 9 |
| 6.2.4 | Test Plan | 9 |
| 6.2.5 | Experimental Results | 10 |
| 7 | Suggested Changes | 10 |
| 8 | Source Code and Deployment instructions | 10 |
| 9 | User Manual | 11 |
| 10 | Lessons Learned | 11 |

List of Figures

| | | |
|---|---|---|
| 1 | Conceptual model of web app | 4 |
| 2 | UI mockup for project web app. | 6 |
| 3 | Diagram of data flow and shape through neural network | 8 |

1 Introduction & Problem Statement

Photoshop is a raster graphics editor used by many creative professionals in their day to day work. For this reason, proficiency with it is a key skill in various professions, and many aspiring creative professionals seek to learn how to use it, often through reading online tutorials.

However, Photoshop tutorials often describe actions using everyday language that does not correspond with specific steps to be taken in the software, which can cause difficulties for novice Photoshop users and affect their productivity. Luckily, recent advances in natural language processing technology, including neural network architectures such as long short-term memory (LSTM) [Huang et al., 2015] and transformers [Devlin et al., 2019] potentially allow for plain text descriptions of Photoshop actions to be translated into their corresponding software commands, otherwise known as the "creative procedural knowledge" required to complete a task. This project intends to build a web app that provides a simple and convenient way for Photoshop users of all skill levels to extract creative procedural knowledge from potentially vague Photoshop tutorials.

2 Literature Review

This project iterates on work done by researchers at Cornell University, ByteDance AI Lab, and Adobe Research as described in their 2019 paper *Creative Procedural-Knowledge Extraction from Web Design Tutorials*. This project will use the same training data, but change the model architecture used for sentence labeling, from FastText to an LSTM-based model. This newer model should outperform FastText's Bag of Words (BOW) based model for reasons that will be elaborated on in the systems level chapter of this report. By doing so, it is hoped that this project will find solutions to the challenges presented in the paper, such as detecting implicit and latent mentions of commands within tutorial sentences. [Yang et al., 2019].

3 Project Risks

There are no physical risks involved in this project. There are cybersecurity risks involved in maintaining a public web application, and care will be taken to choose secure passwords and implement the principle of least privileges

when managing access permissions between different components of the web app. There is also a financial risk, since the pay-as-you-go model used by many web hosting services makes clients potentially susceptible to cost overruns if they experience an unexpected rise in web traffic. This can be mitigated with the use of a prepaid debit card.

4 Use Cases

The main use case for this project is that of a novice Photoshop user following along with an online tutorial with the intention of completing a certain task. This user may lack the knowledge and experience required to identify the Photoshop command that corresponds with a task that the tutorial instructs them to perform.

5 Conceptual Model

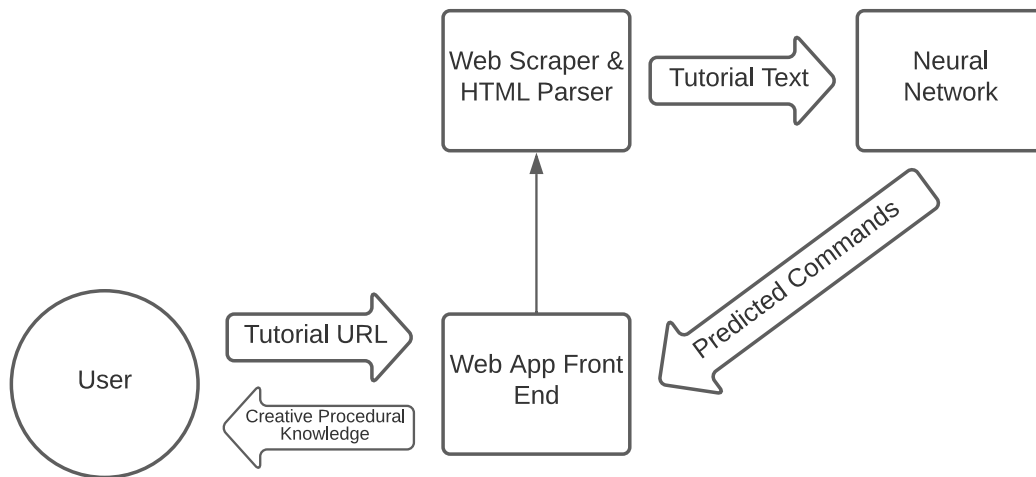


Figure 1: Conceptual model of web app

Conceptually, the project consists of a website where the user is prompted to input a text snippet from a Photoshop tutorial. This text will then be pre-processed and sent as input to the accompanying neural network, which will predict which commands correspond to each sentence of the tutorial and display them to the user.

6 Systems Level Chapter

6.1 User Interface

This section will focus on the final web app’s user interface—that is, the means through which the user interacts with and receives output from the other elements of the project.

6.1.1 Requirements

In terms of functional requirements, the project features a user interface that:

- Prompts the user to input a Photoshop tutorial URL
- For each sentence in the given Photoshop tutorial, displays a list of predicted commands that are referred to in that sentence

When it comes to non-functional requirements, the user interface additionally:

- Display confidence percentages alongside predicted commands, to give the user a sense of how accurate the neural network believes its predictions to be.
- Is simple, responsive, lightweight, and non-resource intensive.
- Has large, pronounced UI elements.
- Has a linear workflow.

6.1.2 UI Mockup

The web app's UI will consist of three main pages: A title page that gives the user basic usage instructions, a loading screen that displays while the inputted text is being processed by the neural network, and a result page with the plain text of the tutorial. Mousing over a sentence will cause it to be highlighted and its command predictions to be displayed beneath it, with associated confidence values.

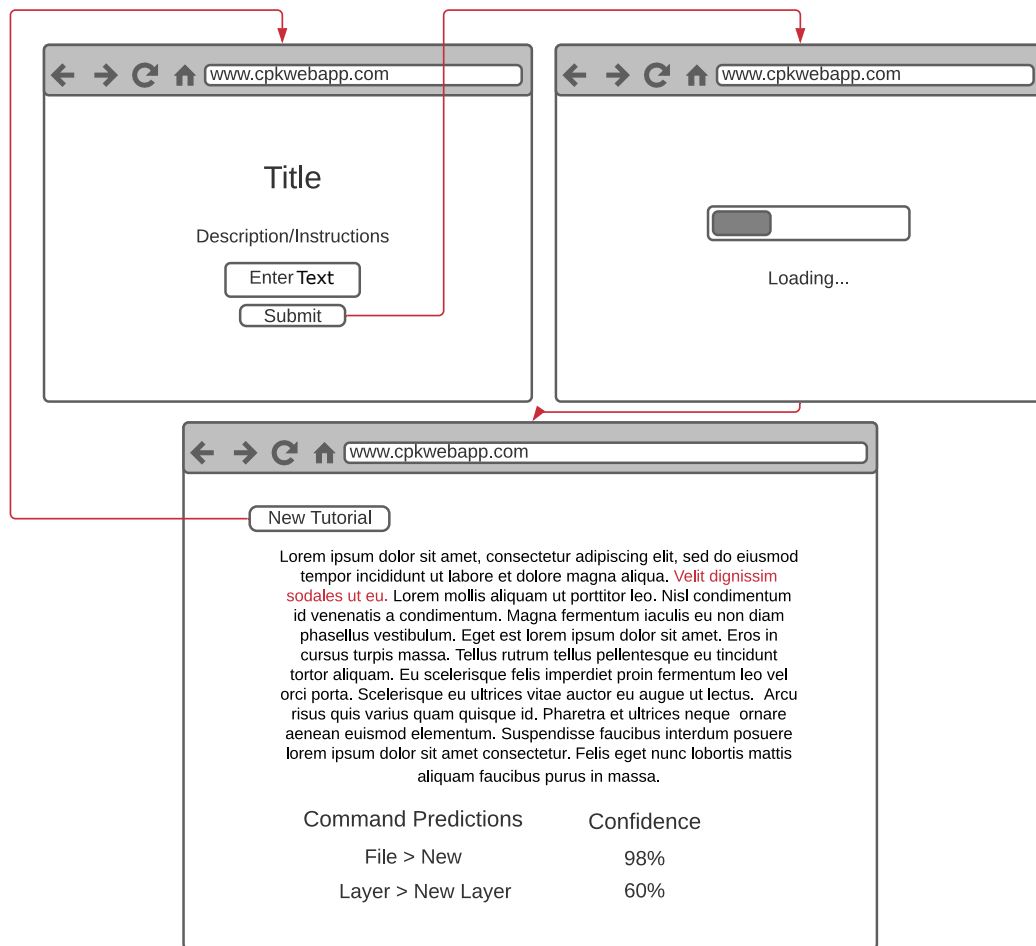


Figure 2: UI mockup for project web app.

6.1.3 Design Rationale

The user interface and the way the user interacts with it will be designed to be simple. The red lines shown in 2 highlight the UI's linear workflow. A simple UI is easier to debug and less prone to failure that would deny the end user access to the neural network's functionality. Ease of use and large, pronounced UI features will also help ensure accessibility for users with visual or motor disabilities.

6.1.4 Technologies Used

The web framework used for the project will be Flask. Flask is easy to use, lightweight, and well-documented, making it ideal for this project, where the main focus of development will be the neural network on the back end and not the website itself. The user interface itself will be written in HTML and JavaScript, with HTML being the current standard for static web page templates and JavaScript the standard for scripting dynamic UI elements

6.2 Neural Network

6.2.1 Requirements

The functional requirements for the neural network entail that it must:

- Given a set of annotated sentences from Photoshop tutorials, train and validate itself on them
- Given a plain-text Photoshop tutorial as input, output predicted commands for each sentence.

The non-functional requirements for the neural network entail that it must:

- Have the ability to be trained in a reasonable amount of time (hours, not days) using the hardware available to this author (Nvidia RTX 3070 + AMD Ryzen 7 1800X + 32GB DDR4 RAM)

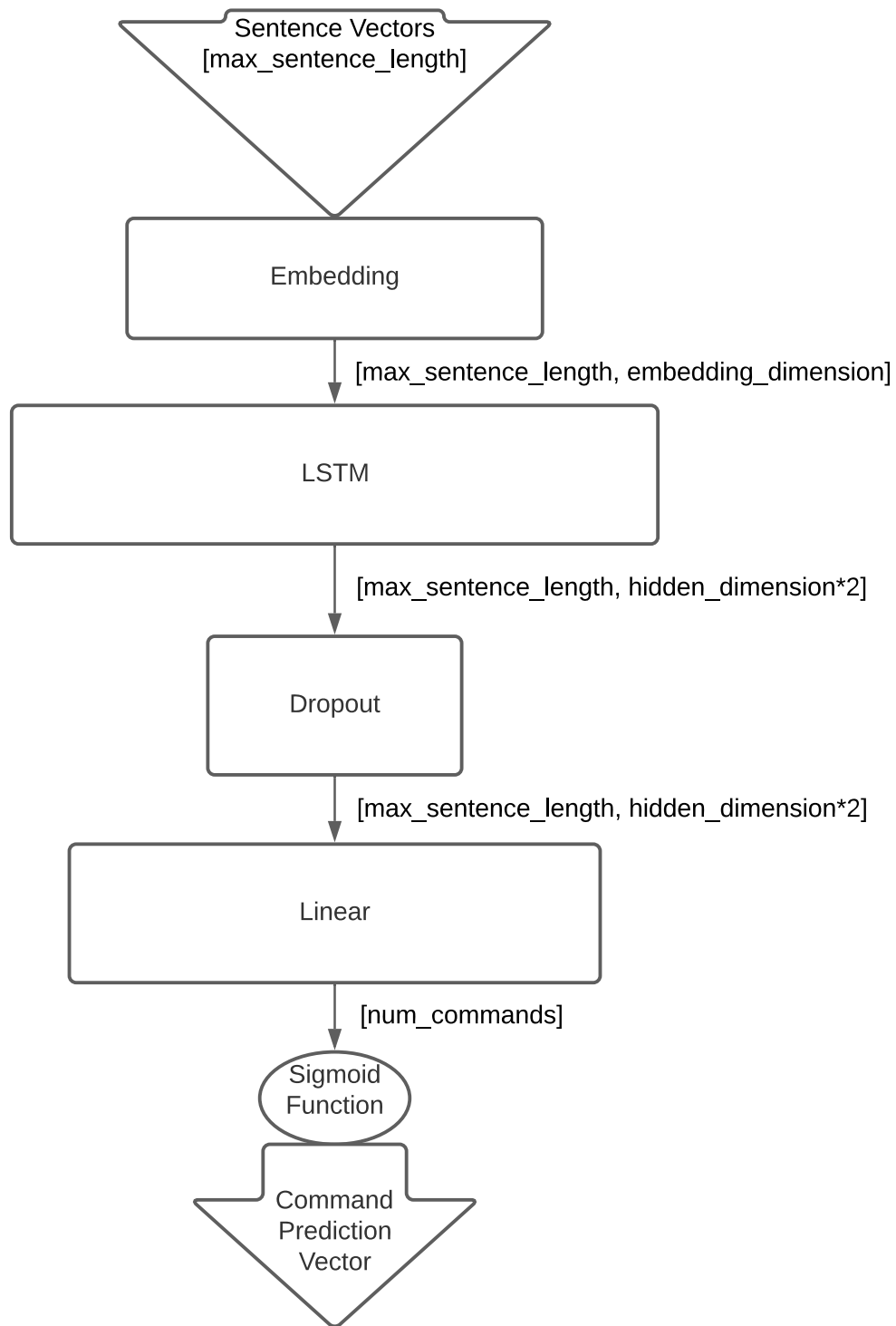


Figure 3: Diagram of data flow and shape through neural network

6.2.2 Design Rationale

As the preliminary component of the network, an embedding layer takes a list of tokens (words) corresponding to each tutorial sentence and converts each token into a high-dimensional word embedding vector, which allows the model to understand the relationships between words and take those relationships into account during training and inference [Chen et al., 2013]. The choice of an LSTM module as the main component of the neural network was due to an LSTM’s ability to remember inputs over time, which should prove useful when dealing with sequential instructions such as those found in Photoshop tutorials [Ding et al., 2018]. The network also features a dropout layer in order to improve the model’s ability to generalize—that is, to perform just as well on unseen inputs as it does its training inputs [Baldi and Sadowski, 2013]. Finally, given that the model is meant to output confidence percentages, a sigmoid activation function is most appropriate, since the sigmoid function will normalize the model’s outputs such that they all lie between 0 and 1 [Narayan, 1997].

6.2.3 Technologies Used

The neural network is written in Python 3.7, using PyTorch, as it is (in the experience of this author) more stable and easy to use than its main alternative, Tensorflow. Additional libraries used include scikit-learn (for dataset splitting and model evaluation), pandas, and optuna (for hyperparameter optimization). Monitoring of training and evaluation metrics will be done through Tensorboard, with a publicly accessible dashboard hosted on Tensorboard.dev. Model training will be performed locally on an Nvidia RTX 3070 GPU.

6.2.4 Test Plan

A thorough grid search was performed over different combinations of the following model hyperparameters:

- Learning rate [1e-4 - 0.01]
- Dropout rate [0.2 - 0.6]
- Batch size [4, 8, 16, 32]

Additionally, the model was tested with the following word embedding schemes:

- GloVe Wikipedia, pre-trained
- DistilBERT, contextual, sum of last 4 hidden layers
- MobileBERT, contextual, sum of last 4 hidden layers

Finally, an additional round of hyperparameter tuning was performed using a model with no LSTM, simply a linear layer on top of DistilBERT.

6.2.5 Experimental Results

Optimal macro average F1 scores for each model tests were as such:

- Pure BERT model: 8%
- LSTM with GloVe embeddings: 13.1%
- LSTM with DistilBERT embeddings: 14.2%
- LSTM with MobileBERT embeddings: 15.1%

7 Suggested Changes

This project might benefit from further measures to help compensate for the neural network's imbalanced training data set. For example, the use of different loss functions, artificial dataset augmentation, or class weighting might all help improve inference results

8 Source Code and Deployment instructions

Source code can be found at <https://github.com/maddenfederico/cpk-webapp>. Deployment can be accomplished using Heroku; its GitHub integration allows for automatic one-click deployment.

9 User Manual

The operation of the CPK extraction tool is quite simple. At entry, the user is prompted with a large text box and a submission button. The user should copy the tutorial text that they want to extract creative procedural knowledge from and paste it into the box, then press the submission button. The user will then be redirected to a page where the original tutorial text will be shown. The user can mouse over different sentences in the tutorial text, which will display for each sentence the top three commands that the neural network predicts to be referenced in said sentence, along with the associated confidence values. At any time, if the user wishes, they can click a button in the top-left section of the page to start over with a different tutorial text excerpt.

10 Lessons Learned

There were a few lessons I learned through the course of this project. The first was that visualization of your dataset at every stage of the training pipeline is important. There were a few times where bugs that would have been obvious had I kept my eye on my dataset went unnoticed for long periods of time. The second would be that, in a neural network-focused project like this one, you should always plan to have something else you can do while your network is training, since training can take anywhere from hours to days.

References

- [Baldi and Sadowski, 2013] Baldi, P. and Sadowski, P. J. (2013). Understanding dropout. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 26, pages 2814–2822. Curran Associates, Inc.
- [Chen et al., 2013] Chen, Y., Perozzi, B., Al-Rfou, R., and Skiena, S. (2013). The expressive power of word embeddings.
- [Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

- [Ding et al., 2018] Ding, Z., Xia, R., Yu, J., Li, X., and Yang, J. (2018). Densely connected bidirectional lstm with applications to sentence classification.
- [Huang et al., 2015] Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional lstm-crf models for sequence tagging.
- [Narayan, 1997] Narayan, S. (1997). The generalized sigmoid activation function: Competitive supervised learning. *Information Sciences*, 99(1):69 – 82.
- [Yang et al., 2019] Yang, L., Fang, C., Jin, H., Chang, W., and Estrin, D. (2019). Creative procedural-knowledge extraction from web design tutorials.






Extracting_Creative_Procedural_Knowledge_from_Photoshop_Tutorials

Final Audit Report

2021-06-11

| | |
|-----------------|--|
| Created: | 2021-06-11 |
| By: | Bioengineering Department (bioengineering@scu.edu) |
| Status: | Signed |
| Transaction ID: | CBJCHBCAABAAyz2LBaQRcL9u8_nVAYmxN4IL4mt91yRa |

"Extracting_Creative_Procedural_Knowledge_from_Photoshop_Tutorials" History

-  Document created by Bioengineering Department (bioengineering@scu.edu)
2021-06-11 - 5:51:00 PM GMT- IP address: 24.6.105.116
-  Document emailed to Nam Ling (nling@scu.edu) for signature
2021-06-11 - 5:51:36 PM GMT
-  Email viewed by Nam Ling (nling@scu.edu)
2021-06-11 - 5:51:38 PM GMT- IP address: 72.14.199.24
-  Document e-signed by Nam Ling (nling@scu.edu)
Signature Date: 2021-06-11 - 5:56:07 PM GMT - Time Source: server- IP address: 75.4.202.62
-  Agreement completed.
2021-06-11 - 5:56:07 PM GMT