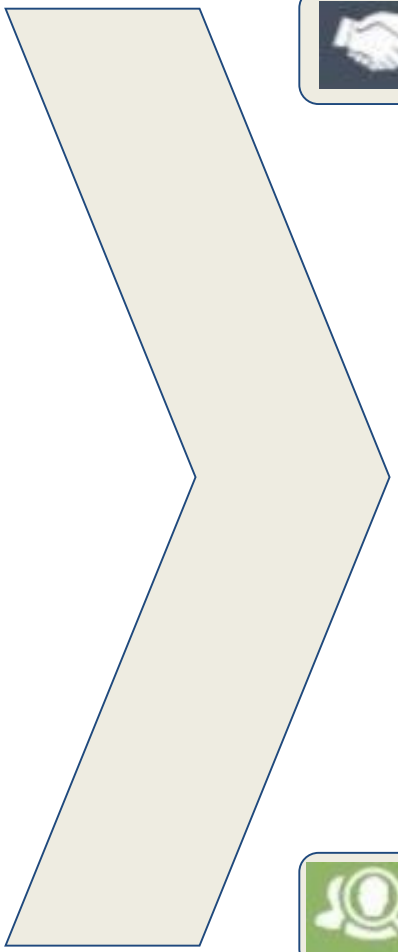
Zorus URL Classification Engine

Team 1:
John G, Anant K, Huanhuan L, Spandana L, Akshay P, Visaga R, Yiliang R

UCONN

Agenda



- Introduction
- Problem Statement
- Solution Overview
- Data Exploration & Pre-processing
- Modeling Methods
- Analysis & Conclusions
- Questions & Answers

Introduction

- A custom classifier will provide flexibility to further **improve Zorus product** and **help to align** with Zorus current technology stack.
- Built a URL classification model that can accurately classify URLs by expanding the current dataset and introducing new variables that affect target variable.
- The model have the ability to classify websites based on **web page content**.
- Our URL Classifier would be a **cost-effective solution** for Zorus and will eventually out-phase the third party classifier as it gets mature.

ZORUS

UCONN

Problem Statement

- The content classification engine provided by third party is a **critical part** of Zorus offering and is a **major risk** to its business model.
- MSPs struggle to provide **cost effective and capable solutions**.
- Margin of error* in classifying URLs would **hinder Zorus product credibility**.

*Misclassification rate

UCONN

Solution Overview

- Performed **web scraping** of URLs to extract complete web page content from websites for analysis.
- Implemented **Natural Language Processing** and **Machine Learning methods** to build the model.
- Our best model, Logistic Regression, is capable to predict “Adult” category with **85%** F1-score.
- To **reduce the misclassification rate by 8.24%** for Adult category we have build a CNN model using image scraping.

Data Exploration

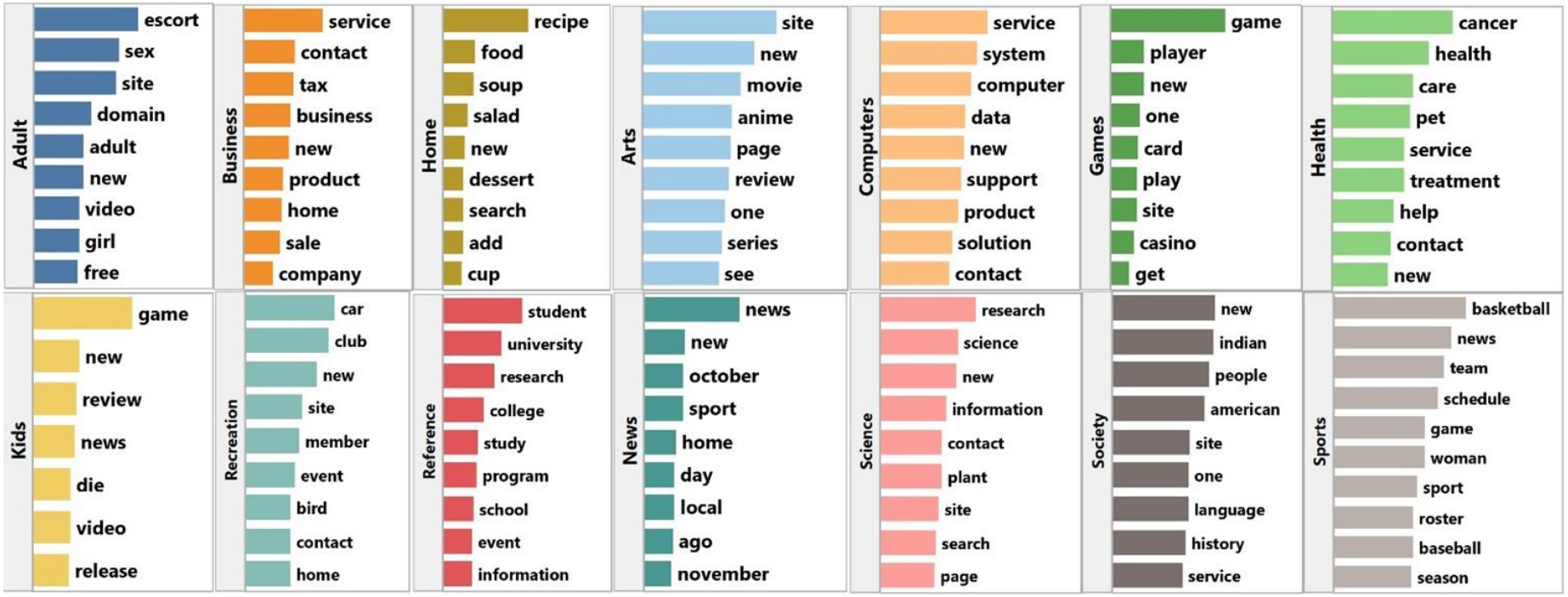


Fig. Most frequent words per category

- We have considered **top 2500 frequent words** per category as input to our Logistic model.

UCONN

Data Exploration

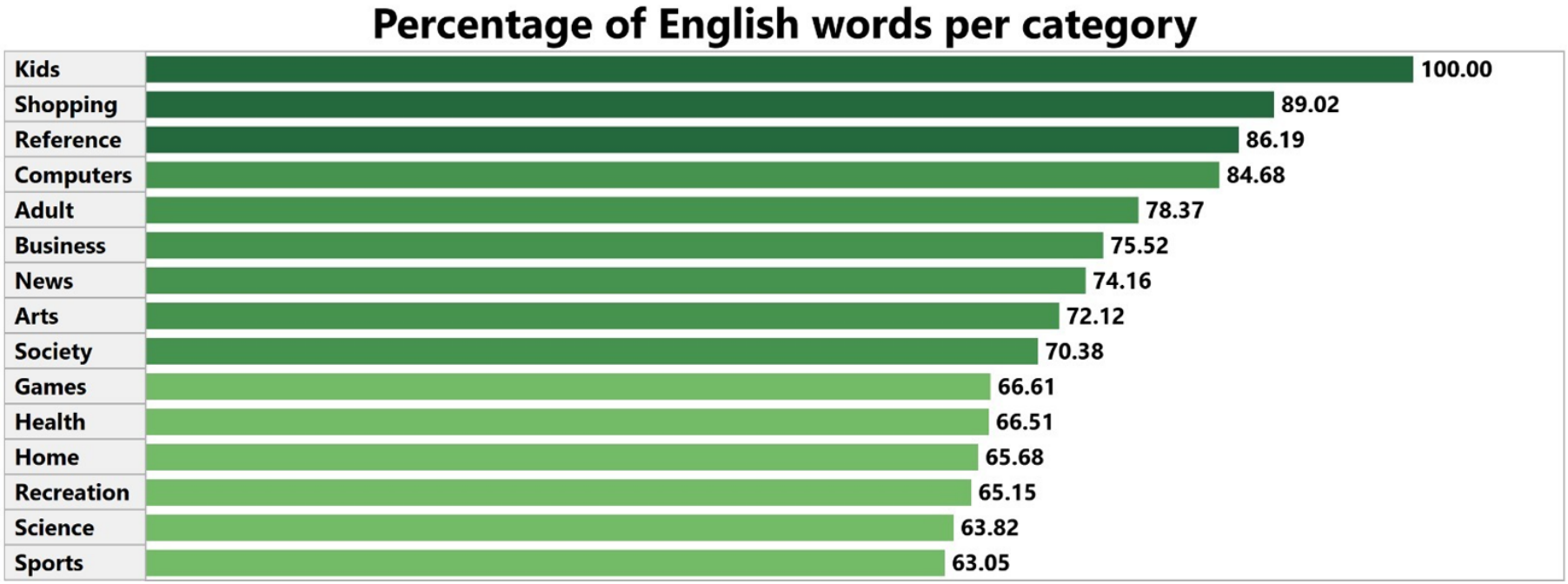


Fig. Percentage of English words per category

- URLs with **Kids category had 100%** of English words and URLs with **Science and Sports category had lowest ~63 %** of English words.
- Non-English words are converted to English words after Web Scraping.

UCONN

Data Exploration

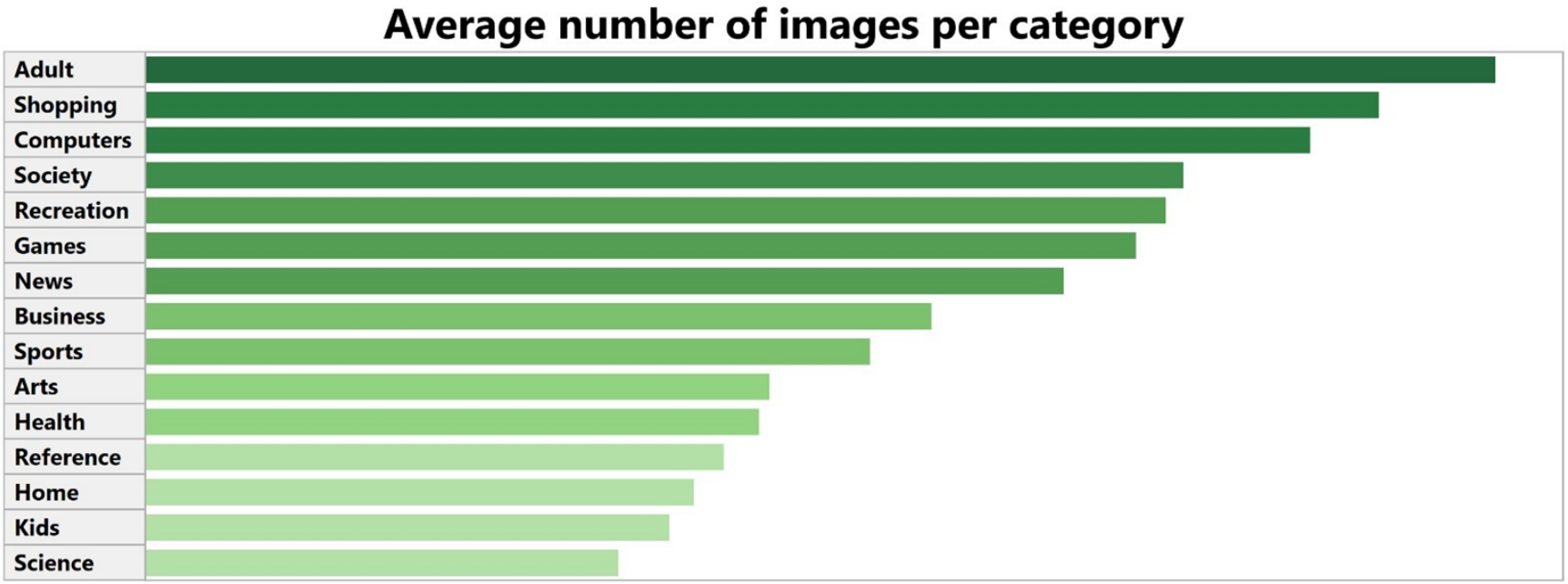


Fig. Average number of images per category

- **Adult** URLs are having **highest** average number of images and **Science** category have the **lowest.**
- The images downloaded by image scraping will be used as an input to CNN model.

Data Pre-processing

- **Excluded inaccessible links** and kept code 202 as valid link.
- Extracted a **balanced** dataset* (10k per category) from given dataset.
- **Top 2500** most frequently used words per category were used to build the logistic regression model.
- To build CNN model, **1k images** for Adult category were downloaded from given dataset and 1k images downloaded for non-adult category from dataset.

Fig. An SEO's Guide to HTTP Status Codes from https://moz.com/blog/an-seos-guide-to-http-status-codes

*Equal number of entries per category

UCONN

Data Pre-processing (Web Scraping)

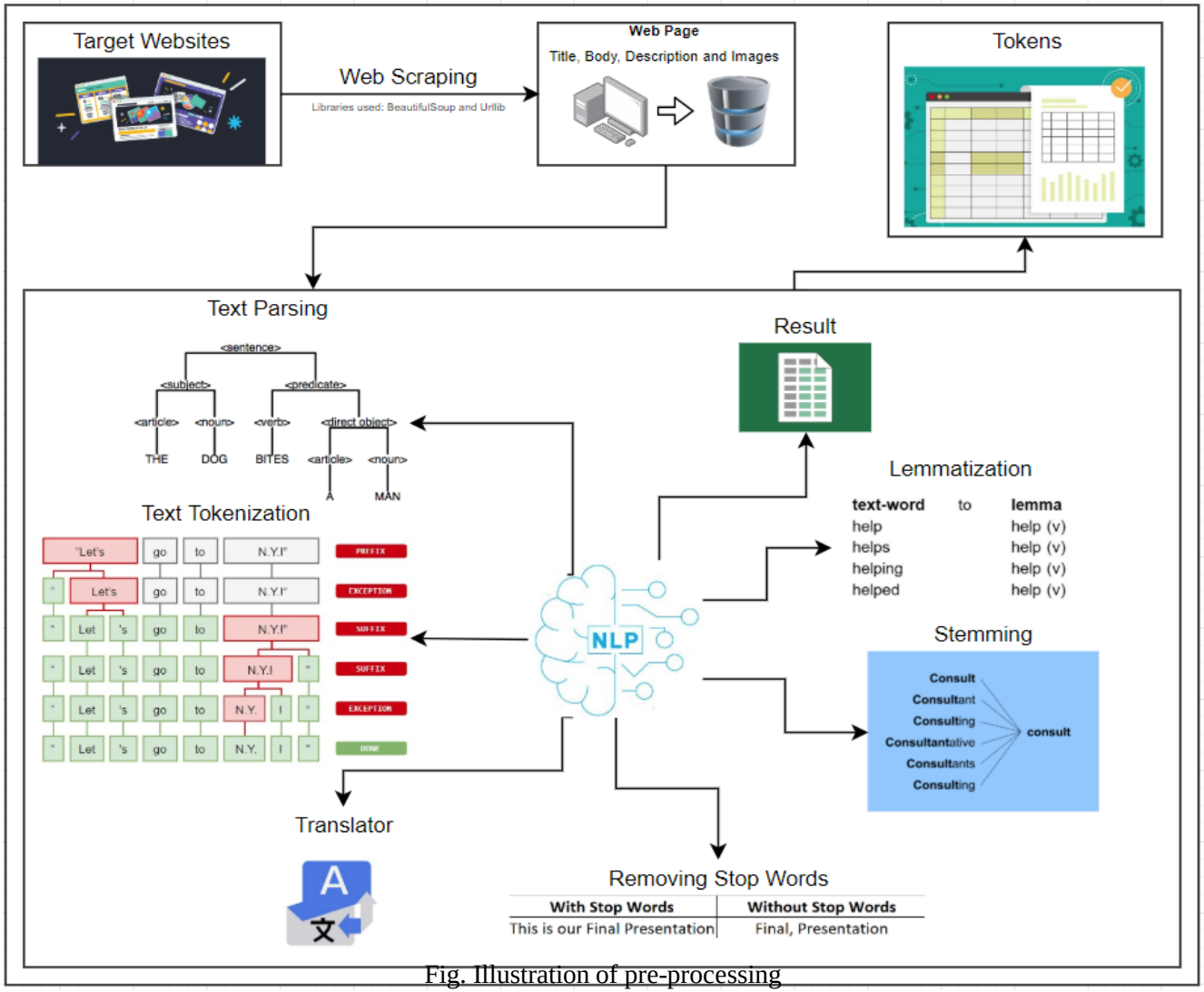


Fig. Illustration of pre-processing

UCONN

Modeling Methods- Logistic regression

- Inputs: Top 2500 most frequent words per category
- Output: 15 categories
- Data Partition: 70% training, 30% validation
- Best Model: Logistic Regression
  - Overall F1-score* : 82%
  - Most Accurate Category: Home (91%)
  - Least Accurate Category: News (77%)
  - Adult Category F1 score: **85%**

Category	Precision	Recall	F1-score
Home	0.91	0.91	0.91
Sports	0.90	0.88	0.89
Kids	0.88	0.88	0.88
Health	0.86	0.85	0.85
Adult	0.80	0.91	0.85
Reference	0.86	0.80	0.83
Science	0.84	0.80	0.82
Business	0.78	0.80	0.79
Games	0.8	0.78	0.79
Recreation	0.79	0.79	0.79
Society	0.80	0.78	0.79
Arts	0.74	0.85	0.79
Computers	0.80	0.76	0.78
Shopping	0.83	0.73	0.78
News	0.83	0.72	0.77
accuracy			0.82
macro avg.	0.83	0.82	0.82
weighted avg.	0.83	0.82	0.82

Fig. Performance metrics for Logistic Regression model

*Precision: Ratio of number of URLs correctly categorised to the total number of URLs categorised

Recall: Ratio of number of URLs correctly categorised to the total number of URLs

F1-score: A weighted average of the precision and recall of a model, with results near to 100% being best and near to 0% being worst

Modeling Methods CNN for Image Classification

- CNN model is built to make sure a URL which is not classified as Adult does not pass the restrictions the business is planning to apply.

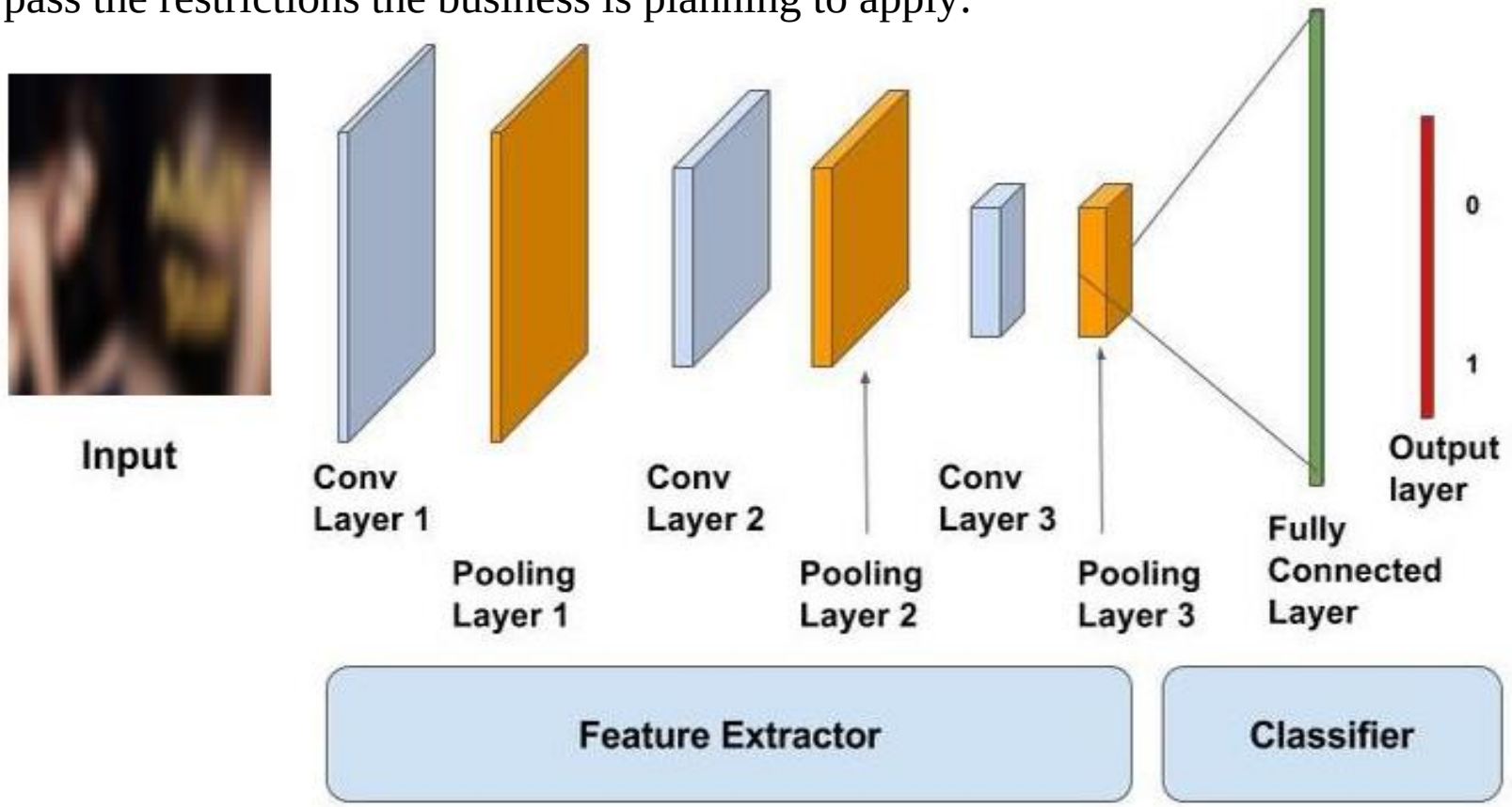
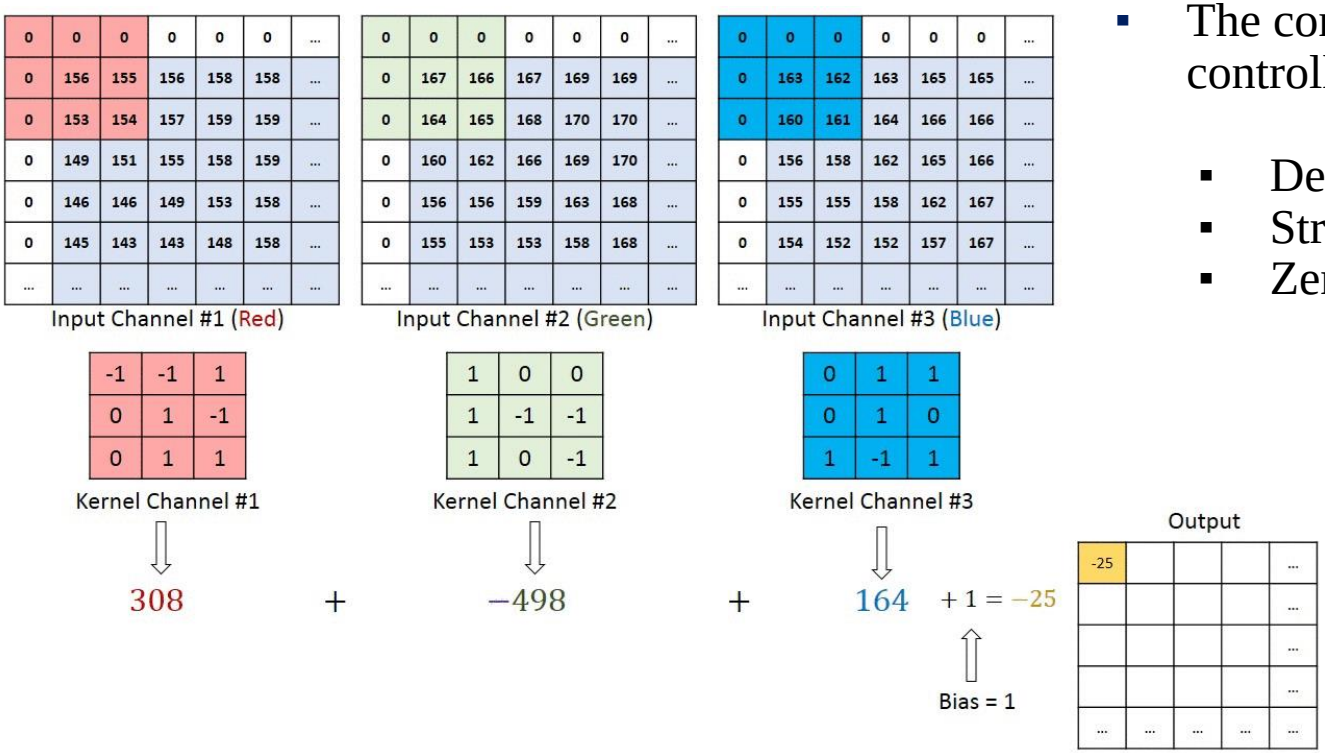


Fig. Architecture of the Convolutional Neural Network

UCONN

Modeling Methods CNN for Image Classification

- **Arithmetic behind the convolution** : This process will apply a filter on a small array of pixels within the picture. The filter will move along the input image with shape of 3x3.
- The output matrix is the result of the element-wise operation between the image matrix and the filter.



- The convoluted features are controlled by 3 parameters:
  - Depth
  - Stride
  - Zero-padding

UCONN

Modeling Methods- CNN

- Data Partition: 70% training, 30% validation
- Input: 1k images of adult & 1k images for non-adult
- Output: 2 categories (adult/non-adult)
- Results:
  - Overall accuracy: **80%**
  - Adult accuracy: **93%**

Model Application & Performance

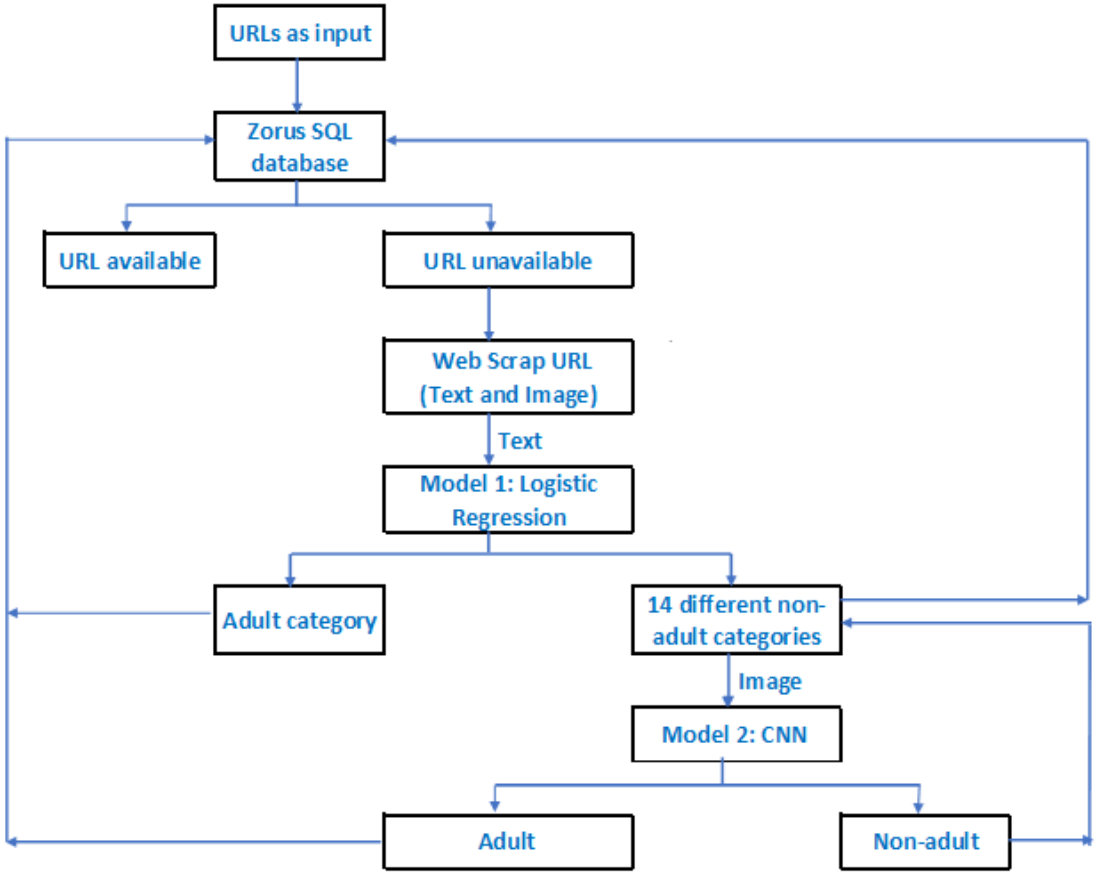


Fig. Proposed implementation flow chart of our model

		Actual	
		Adult	Non-Adult
Predicted	Adult	TP	FP
	Non Adult	FN	TN

After Logistic
TP+TN = 85%

Before CNN model
FN = 10.3%
FP = 4.70%
Misclassification rate (FN+FP)=15%

After CNN model (~80% accuracy)
FN decreased by 10.3% * 80% = 8.24%
Misclassification rate (FN+FP)=6.76%

Fig. Performance of models

UCONN

Analysis & Conclusion

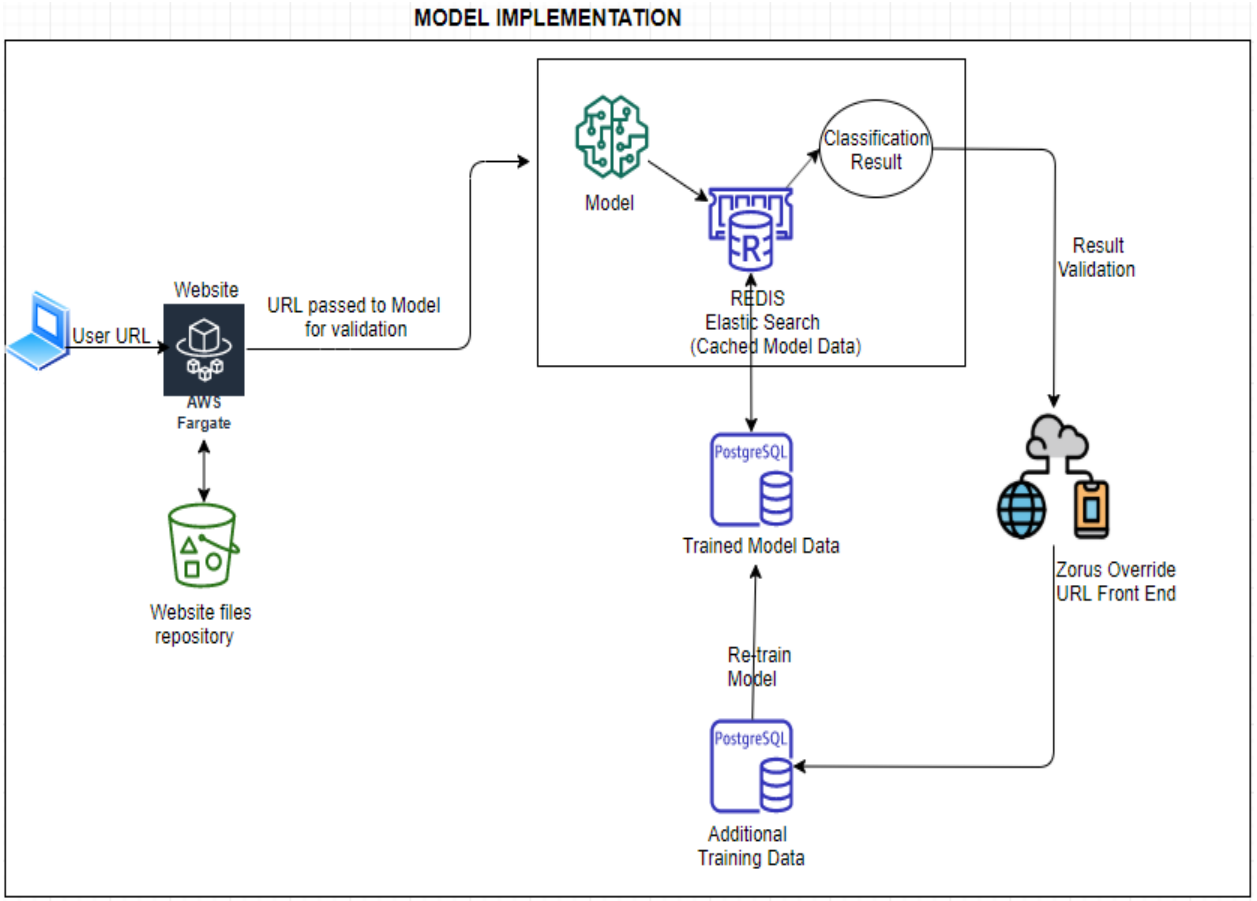
In conclusion,

Model	Total Accuracy	Adult Accuracy	No. of entries in test data	Execution time for test data
Logistic	82%	85%	60k URLs	4 min
CNN	80%	93%	600 images	2 min

- Increase in classified URLs will **build our model's memory** thereby, providing **more accuracy** and better results in future.
- Our model is **cost effective** and we have added an **extra filter** through CNN model to accurately classify adult categories.

Suggestions

We suggest including **AWS** as part of the final solution that will be implemented by Zorus.



To reduce processing time in future, below 6 tools may help to speed up our Python code-

1. **Intel Distribution for Python**
2. **Numba**
3. **swiftapply**
4. **Dask**
5. **Cython**
6. **PySpark**

UCONN

Questions?

Thank You