Regular Expressions & Deep Learning on Picotte Example

Thomas Coard & Bahrad A. Sokhansanj, Ph.D.

Rosen Group – EESI Lab
Drexel University
December 16, 2021

Outline

History and Standards

Basic Syntax

Computation Theory

Useful Examples

History

1951: Stephen Cole Kleene described regular languages using his mathematical notation called regular events.

1970’s: Ken Thompson, the co-creator of unix, put them in the text editors used in unix. This syntax was also used in other unix tools written by other people, such as grep, vi, lex, sed, and AWK.

Grep for instance comes from the text editor ed, where g/re/p meant "Global search for Regular Expression and Print matching lines”

Standards

Almost every language implements regular expressions in its own way, but these are the three most common standards that tools and languages base their regular expressions off of.

In order of complexity:

- POSIX Basic Regular Expressions
- POSIX-Extended Regular Expressions
- PCRE (Perl Compatible Regular Expressions)

Standards

Some tools can utilize different standards. GNU’s version of grep can use all three with the correct flags.

- `grep` uses POSIX Basic Regular Expressions
- `grep -E` uses POSIX-Extended Regular Expressions
- `grep -P` uses PCRE (Perl Compatible Regular Expressions)

Basic Syntax

- `.` Match any character.
- `*` Match preceding character 0 or more times.
- `[]` Match character(s) enclosed in brackets. Can be given a range, such as `A-Z` to match all uppercase letters.
- `[^]` Don’t match character(s) enclosed in brackets. Characters are placed after `^` here.
- `^` Match starting at the beginning of a line.
- `$` Match at the end of a line.
- `\` Escape succeeding special characters so they are treated like a normal character. (Can also make succeeding character have a different meaning).

Practice Example

https://regexr.com/

Write a regular expression that finds all:

- Substrings that are composed of any character followed “ll” (this can be in the middle of a word).
- Finds all substrings starting with “p” and end with “e”.

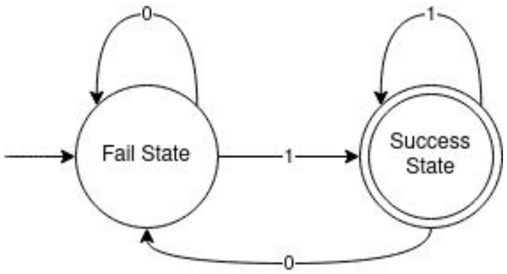
Theory: Finite Automaton

Finite Automaton are abstract machines that are composed of a finite number of states and conditions that trigger state changes.

They are the simplest machines that can recognize patterns. But they do not have all of the capabilities of systems that are Turing complete.

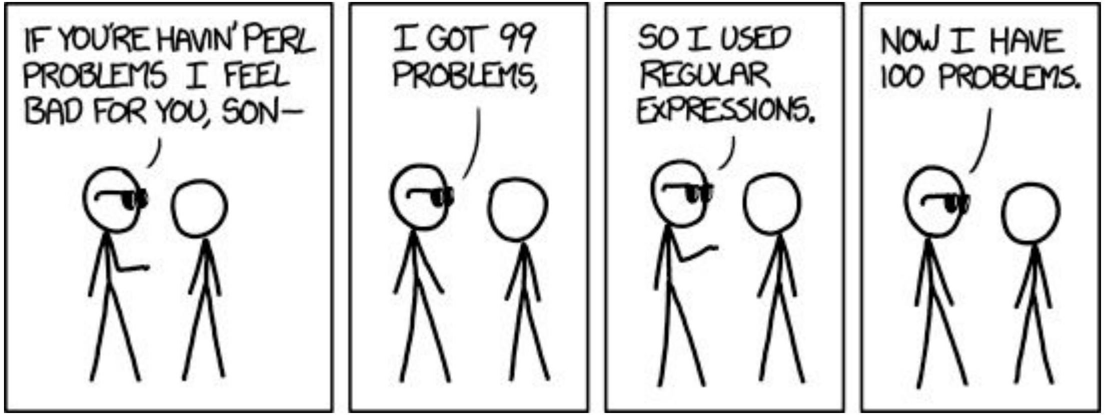
One simple example that cannot be computed by Finite Automaton/Regular Expressions is matching a string that has the same number of 1's and 0's.

[10]*1

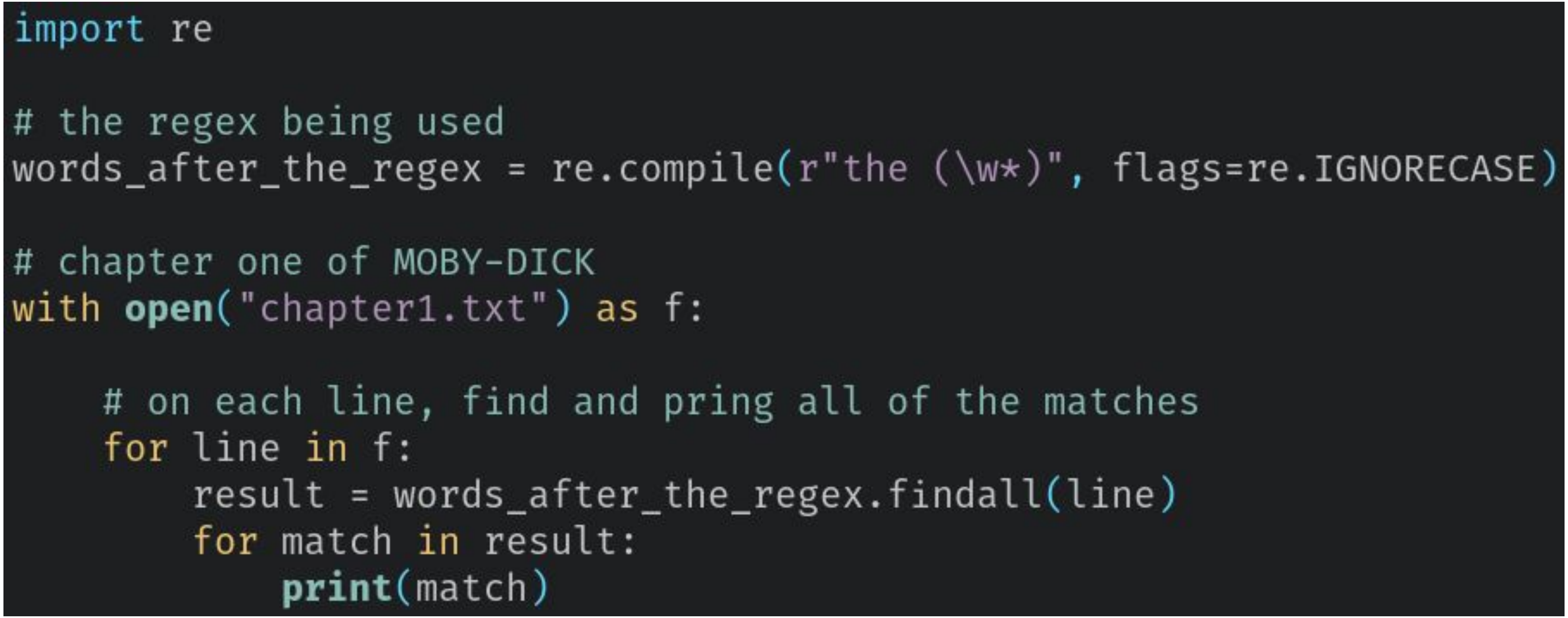


An approachable book on this topic and other areas of computation is: *Turing's Vision The Birth of Computer Science* by Chris Bernhardt.

Example in Python


IF YOU'RE HAVIN' PERL PROBLEMS I FEEL BAD FOR YOU, SON—
I GOT 99 PROBLEMS,
SO I USED REGULAR EXPRESSIONS.
NOW I HAVE 100 PROBLEMS.

https://xkcd.com/1171/



```python
import re

# the regex being used
words_after_the_regex = re.compile(r"the (\w*)", flags=re.IGNORECASE)

# chapter one of MOBY-DICK
with open("chapter1.txt") as f:

    # on each line, find and pring all of the matches
    for line in f:
        result = words_after_the_regex.findall(line)
        for match in result:
            print(match)
```

Linux Command Line Examples

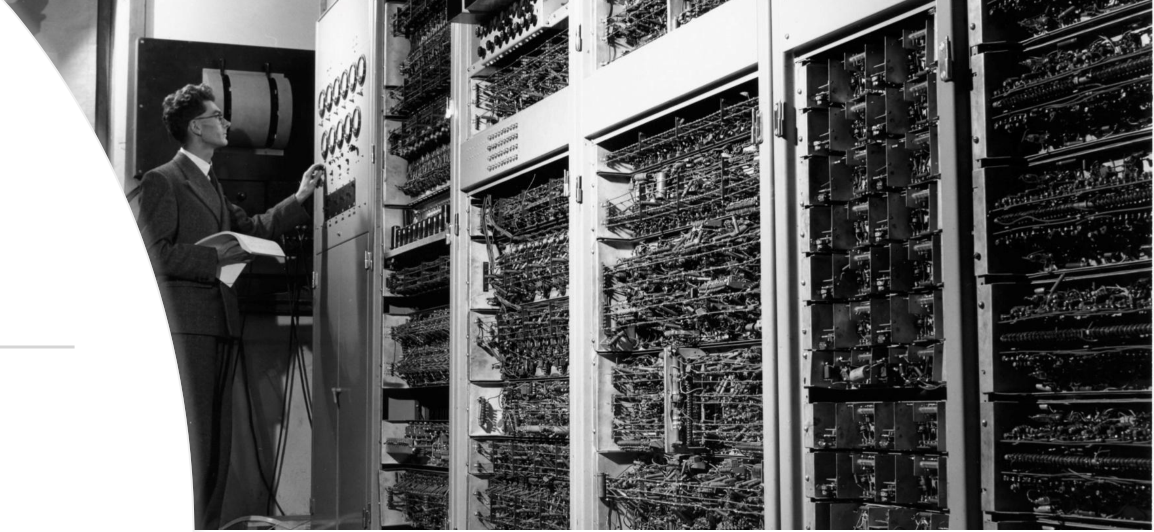
- * in bash
- https://github.com/stephenturner/oneliners

Deep Learning on Picotte Example

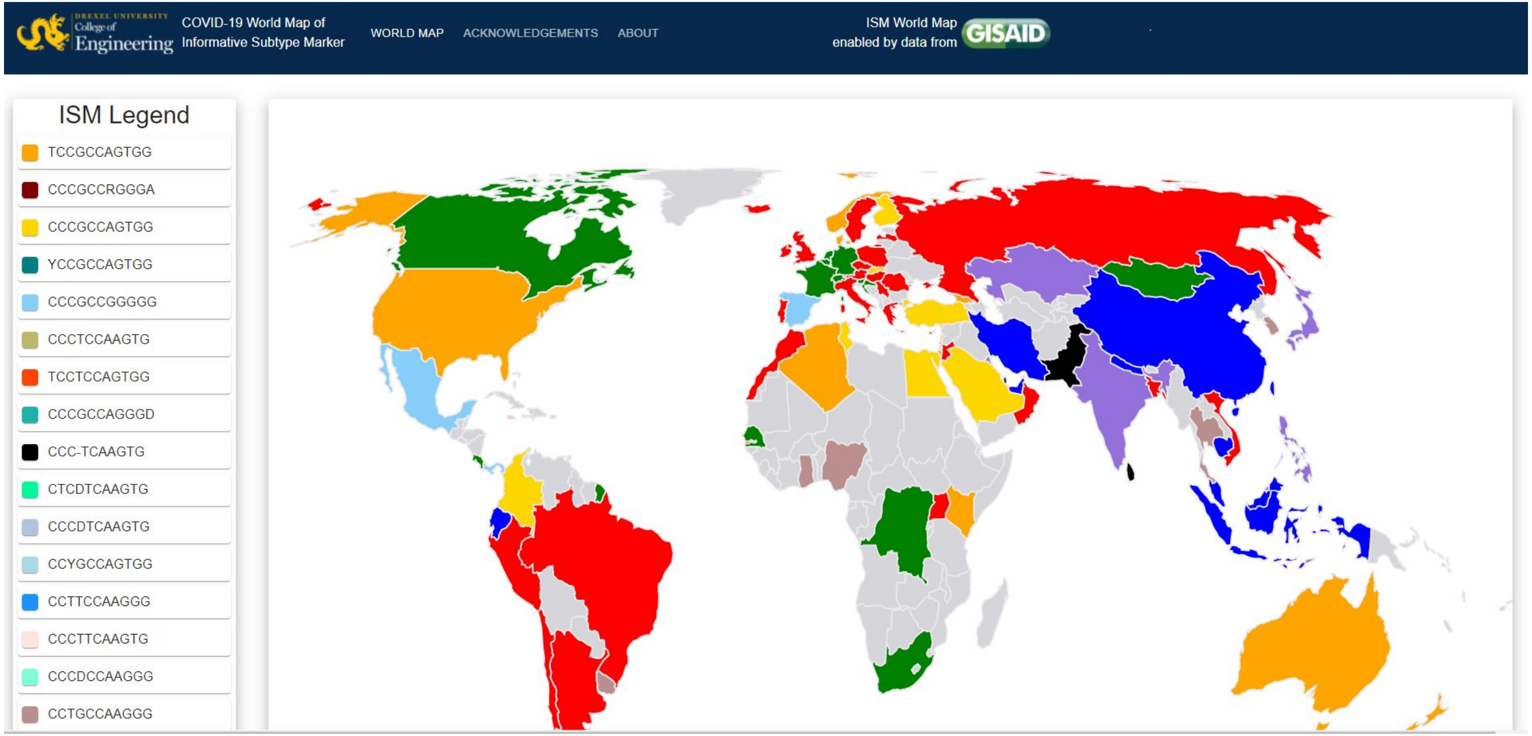
Bahrad A. Sokhansanj, Ph.D.

Rosen Group – EESI Lab
Drexel University

December 16, 2021



COVID-19 (SARS-CoV-2)

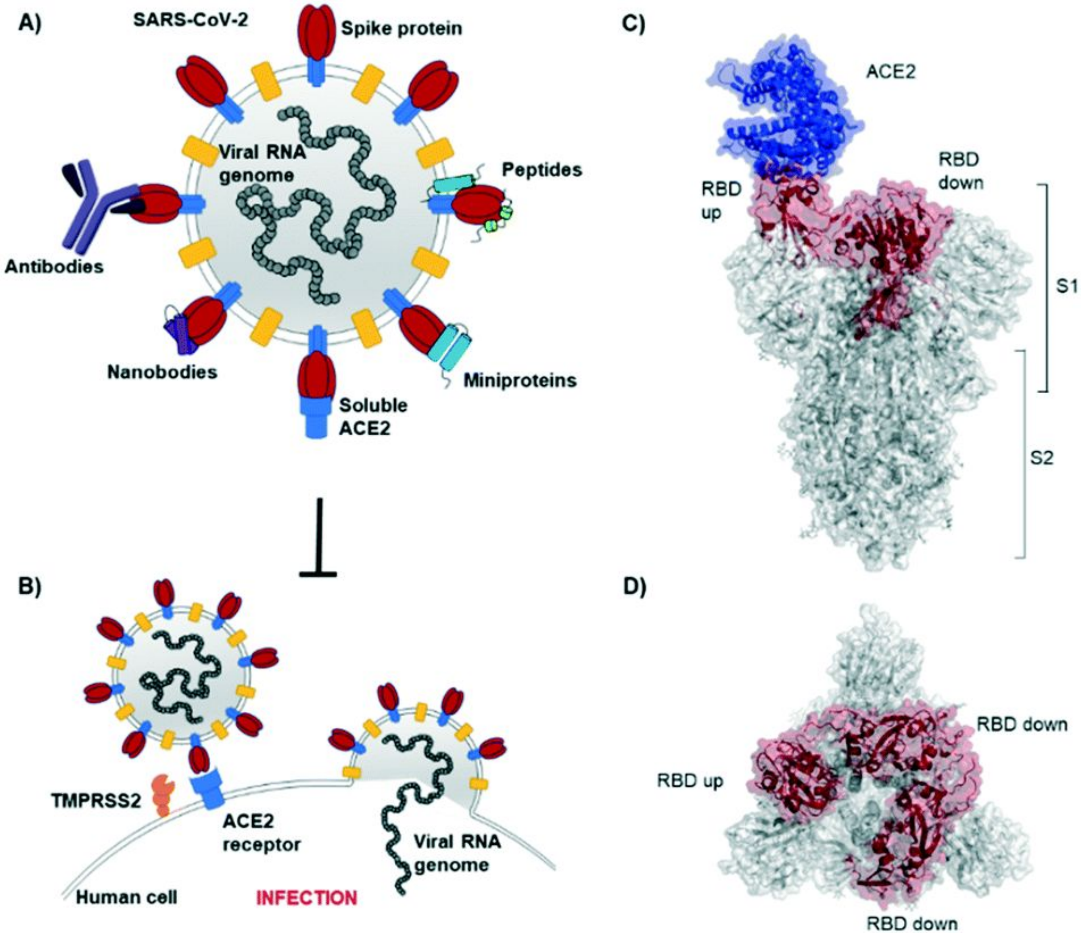


COVID-19 World Map of Informative Subtype Marker

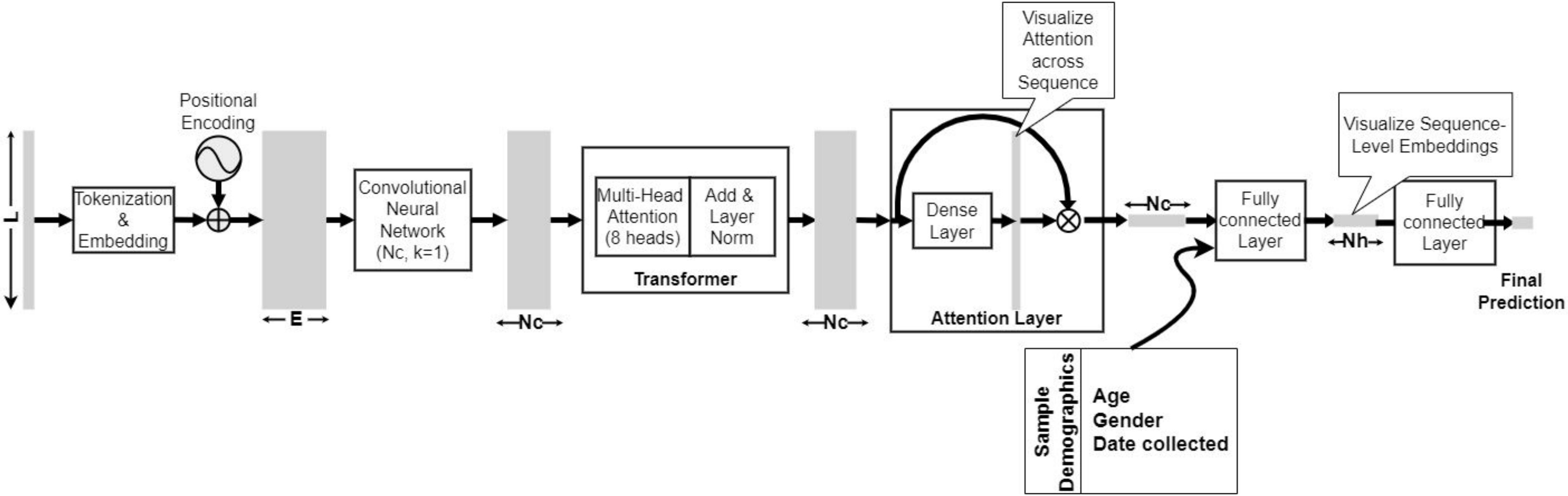
ISM World Map enabled by data from GISAID

ISM Legend

🟧	TCCGCCAGTGG
🟫	CCCGCCRGGGA
🟨	CCCGCCAGTGG
🟦	YCCGCCAGTGG
🟦	CCCGCCGGGGG
🟫	CCCTCCAAGTG
🟧	TCCTCCAGTGG
🟦	CCCGCCAGGGD
⬛	CCC-TCAAGTG
🟩	CTCDTCAAGTG
🟦	CCCDTCAAGTG
🟦	CCYGCCAGTGG
🟦	CCTTCCAAGGG
⬜	CCCTTCAAGTG
🟩	CCCDCCAAGGG
🟫	CCTGCCAAGGG


A)
SARS-CoV-2
Spike protein
Viral RNA genome
Peptides
Antibodies
Nanobodies
Soluble ACE2
Miniproteins
B)
TMPRSS2
ACE2 receptor
Viral RNA genome
Human cell
INFECTION
C)
ACE2
RBD down
RBD up
S1
S2
D)
RBD down
RBD up
RBD down

Model



L
Tokenization & Embedding
Positional Encoding
E
Convolutional Neural Network (Nc, k=1)
Nc
Multi-Head Attention (8 heads)
Add & Layer Norm
Transformer
Nc
Visualize Attention across Sequence
Dense Layer
Attention Layer
Nc
Fully connected Layer
Visualize Sequence-Level Embeddings
Nh
Fully connected Layer
Final Prediction
Sample Demographics
Age
Gender
Date collected

Example of Tensorflow on GPU

- https://proteusmaster.urcf.drexel.edu/urcfwiki/index.php/TensorFlow

- https://proteusmaster.urcf.drexel.edu/urcfwiki/index.php/Slurm_-=_Job_Script_Example_05_TensorFlow_Singularity

Thank you

- For any questions or follow-up, please feel free to contact me directly:
    - Bahrad Sokhansanj, bahrad@molhealtheng.com or bas44@drexel.edu