ML for Text

Data4ML

Summer 2022

Text

- Text analysis utilized for creating algorithms that make prediction based on raw sequences. A small sample:
  - **DeePromoter: Robust Promoter Predictor Using Deep Learning**
    - https://doi.org/10.3389/fgene.2019.00286
  - **DNA sequences performs as natural language processing by exploiting deep learning algorithm for the identification of N4-methylcytosine**
    - https://www.nature.com/articles/s41598-020-80430-x
  - **kmer-SVM: a web server for identifying predictive regulatory sequence features in genomic data sets**
    - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3692045/
- Most of this work is inspired by algorithms for more traditional text analysis
  - You'll find lots of online tutorials for these tools
- We'll talk about both, and the connection between the two.

Reminder What Makes a Dataset AI Ready?

- Clear Data Splits
  - Designate some examples as training data
  - Designate some examples as testing data
- **It contains a clear set of consistent examples**
- The Dataset is Clean
  - No missing values or ‘Nans’
  - No ‘bad’ data
  - **Consistent as Possible**

A Text Dataset Example

Text	**Labels**
“I thought this was a wonderful way to spend time on a too hot summer weekend, sitting in the air con...”	“positive” or 1
“So im not a big fan of Boll's work but then again not many are. I enjoyed his movie Postal (maybe im...”	“negative” or 0
An awful film! It must have been up against some real stinkers to be nominated for the Golden Globe....	“negative” or 0

A Genomics Dataset Example

Text	Labels
CTCCACTTTTTCTCACGTTTATCTGAGCGAAAACAAGCACGGTTCGGCAGCCTCCTT TCCCAGCCCTACCTTTGTGCTGCAAAAGCGAAAATTCAAAAGCCAAGTACAATAGG AGACCGCCCACCCTGGCTCCCTCGTGACACGAGGGAGCGCGAAGCGGAGGGCGC CTCGCGGCAGGAGCGGGATTTCCGGGGTCACGGGAACCGGCAGGGGAACGGGA TAAAGTTCCCGGAGAAAGGAAAGGAGAGCGTGGGATAGTAAAAGAGAAGACGCG GAGAAGAGGAGAGGACCTACAAGAAC	“promoter” or 1
CCAGCAGATGGAAAACAGGACAATGTAACACTGTTCTTATCATCACTATCAGCTGGG ACCAGAACAGACACTCAATAAACAGCCTCACACTACAATGAAGCTTGGAGAACAAA GGAGCATCAAAGGGACATGGAGGGCAAGGGTAGCTCTTCTGCTCCCCAATCACAT GCACTCCCCGTCTCACCGCAACATCTGTCCCTGAGCCTTCTCCCAGCAGACCTATAA ATCCAGGCTGGCTCCTCACTCCCCACACATCTGCTCCTGCTCTCTCTCCTCCAGCGAC CCTAGCCATGAGAACCC	“promoter” or 1
CAAGCAGATGGAACACAGGACAATGTAACACTGTTCTTATCATCACTATCAGCTGGG ACCAGAACAGACACTCCATAAACAGCCTCACACTACAATGAAGCTTGGAGAACAAA GGAGCATCAAAGGGAAATGGAGGGCAAGGGTAGCTCTTCTGCTCCCCAATCACAT GCACTCCCCGGGGCACCGCAATTCAGTCCCTGAGCCTTCTCCCAGCAGACCTATAA ATCCAGGCTGGCTCCTCACTCCCCACACATCTGCTCCTGCTCTCTCTCCTCCAGCGAC CCTAGCCATGAGAACCC	“not a promoter” or 0

A promoter, as related to genomics, is a region of DNA upstream of a gene where relevant proteins (such as RNA polymerase and transcription factors) bind to initiate transcription of that gene. – genome.gov

Unstructured Data - Reminder

- Unstructured Data is generally used to refer to anything that doesn't easily fit into our normal rows are examples columns are variables data model
- Images, **text**, sounds etc.
- Two common ways of dealing with this data
  - **Feature engineering:** Writing methods to extract 'structured data'
    - i.e. A Bag of words from text
  - **Deep Learning**

Feature Engineering

- The goal for featuring engineering is turning unstructured data into structured data, while still capturing meaning.
  - Text snippets can have different lengths which isn't something that can SVM Random Forests, or other structured algorithms can handle.
- A common method is to use a 'bag' of words

  1) Start with a **Vocabulary** – all the words in your dataset

  2) For each snippet of text count the number of times each work appears in your text snippet, this is your structured data

  "**The quick brown fox jumps over the lazy dog**"
  -> {the:1, cat:0,quick:1,brown:1,fox:1,jumps:1,over:1 …}

  Gives you one continuous variable for each word in your vocabulary

Bag of words Things to Watch Out For

- **Vocabulary** – You get to decide what your vocabulary is, it doesn't have to be just one word.
  - Problem: These have very different meaning, but the same bag of words
    - The cat is in the house not in the backyard
    - The cat is in the backyard not in the house
  - Use n-grams or several words in your vocabulary

- **Vocab= {the, cat, is, in, house, not, backyard}**

OR

- **Vocab= {the cat, in the house, in the backyard, not}**

You can tune these lists or you all possible combinations of n-words

- **A gotcha – ML algorithms can easily memorize unique features, so make you have enough examples of each of your vocab words. For example if an n-gram only appears once in your dataset you may want to throw it out or included in a special 'other' count.**

K-mers

- Same principal is used for sequences
  - 1-mer = {A,T,G,C}
  - 2-mer = {AA,AT,AG,AC,
    TA,TT,TG,TC,
    GA,GT,GG,GC
    CA,CT,CG,CC}
  - etc…
- Unlike n-grams in works these are often calculated with overlap
  - i.e. “ **This Sentence**” {This:1, Sentence:1, …} no overlap
  - VS “ATGC” = {AT:1, TG:1, GC:1, …} with overlap

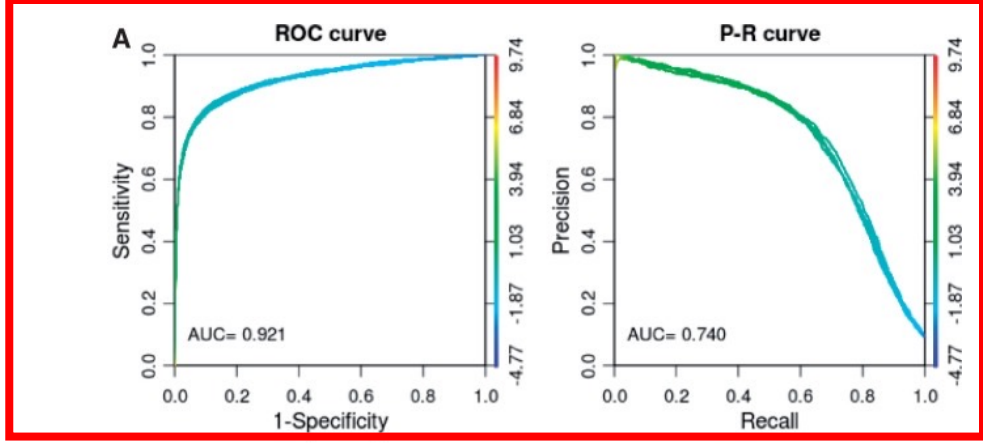
A 1-mer bag of words doesn’t provide much information a 10-mer has so many options that you may not have many repeated elements, it’s a balance that depends on dataset size.

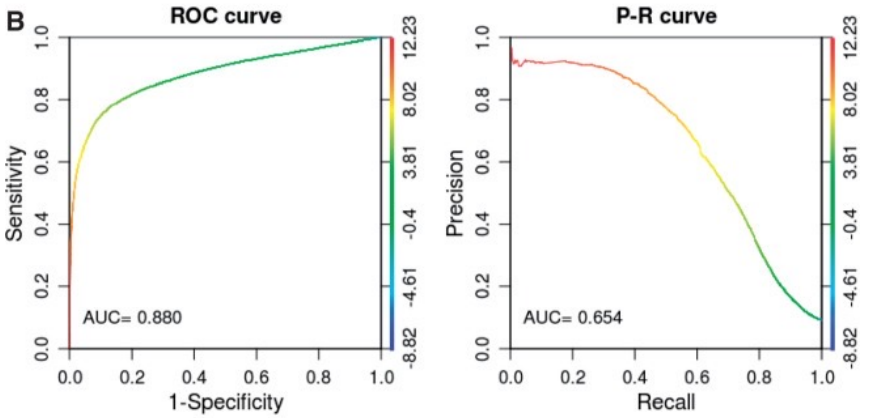
Feature Engineering Workflow

- Gather text examples into an ‘AI ready’ dataset
  - Make sure you have train and test set
- Define a vocabulary
  - Pick a k/n and make sure you it covers your training and testing sets, and make sure all n-grams or k-mers appear at least a few times in the both datasets
  - Require more appearances or reduce n/k if you have trouble with overfitting
- Convert your text into a count vector
- Feed count vectors into your favorite ML tool
  - SVM, Random Forest
  - TSNE, PCA, UMAP

Example **kmer-SVM**

- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3692045/
- Looking ESRRB bound regions trying to identify common patterns in a 100bp window
- Negative samples selected to match distributions of length and GC content in the Positive (ESRRB) sites
- Identified new motifs, showing that A or G is allowed in the binding site at the 11th position





C

6-mers	Revcomp	SVM Scores
Positive 6-mers		
AAGGTC	GACCTT	10.05
AGGTCA	TGACCT	8.47
ACCTTG	CAAGGT	5.33
AGGTCG	CGACCT	5.17
GGTCAA	TTGACC	4.01
Negative 6-mers		
GCAATA	TATTGC	-2.05
TGACCA	TGGTCA	-3.33
AAGGTA	TACCTT	-4.23
AGACCT	AGGTCT	-4.55
AGGTCC	GGACCT	-4.98

D

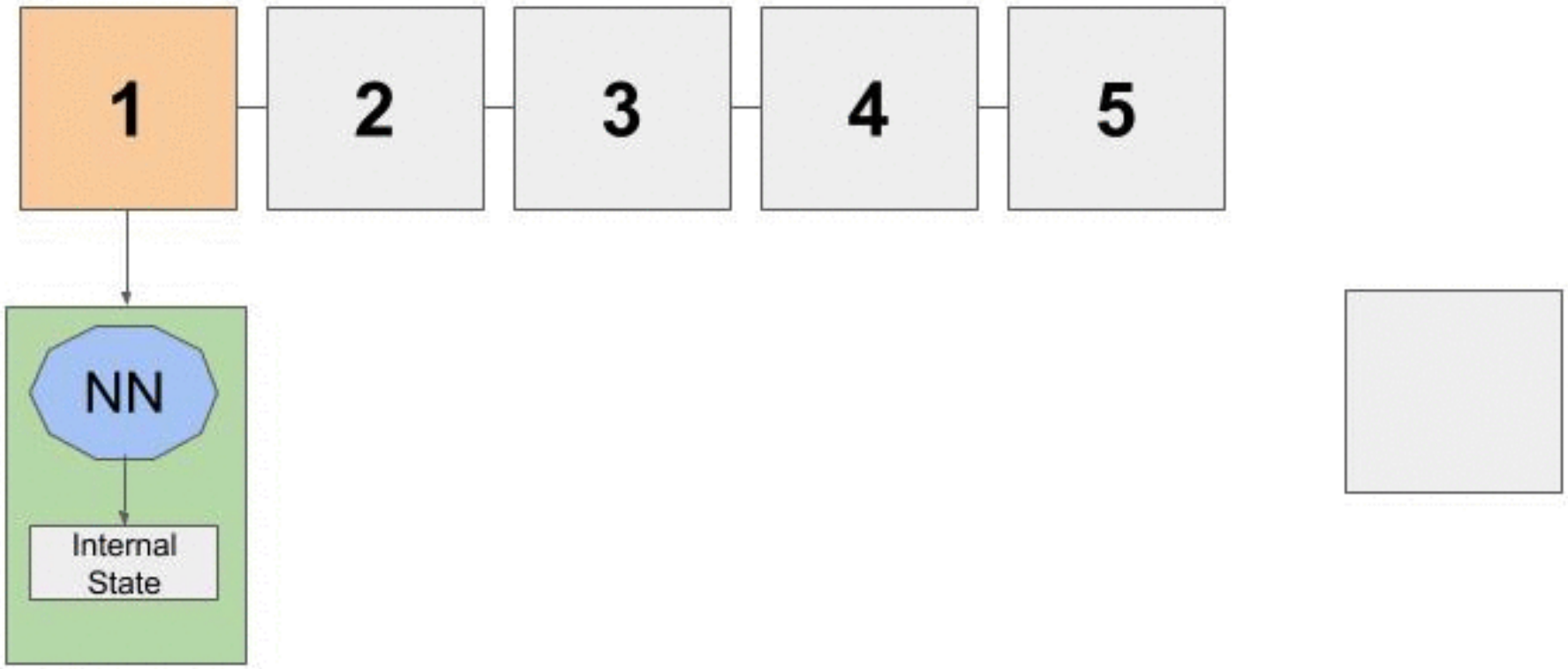
Deep Learning

- Unlike the above algorithms Deep Learning doesn't neatly fit into an R packages
- It’s better to think of it as a box of Lego blocks
- Deep Neural networks are made up of layers
- Layers exist for text, images, structured data, etc.
- Allows for building and training custom models
- This models are often made available

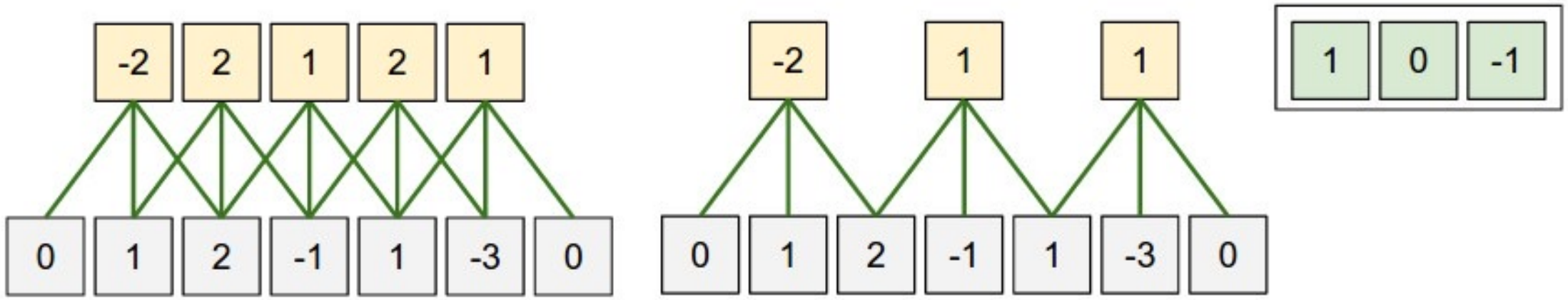
Deep Learning Series Data

- A number of layers/models used in deep learning are designed process data of variable length like text, some common ones are
  - LSTM – Stand for Long Short Term Memory, and is a common method of processing data in order one token (work/n-gram/k-mer) at a time. Slow.
  - CNN – Also used in images processes nearby elements in a sequence. Fast, but not as good at long range interactions.
  - Transformers – Quickly becoming the most common deep learning algorithm for text. Computationally expensive.

LSTM



CNN

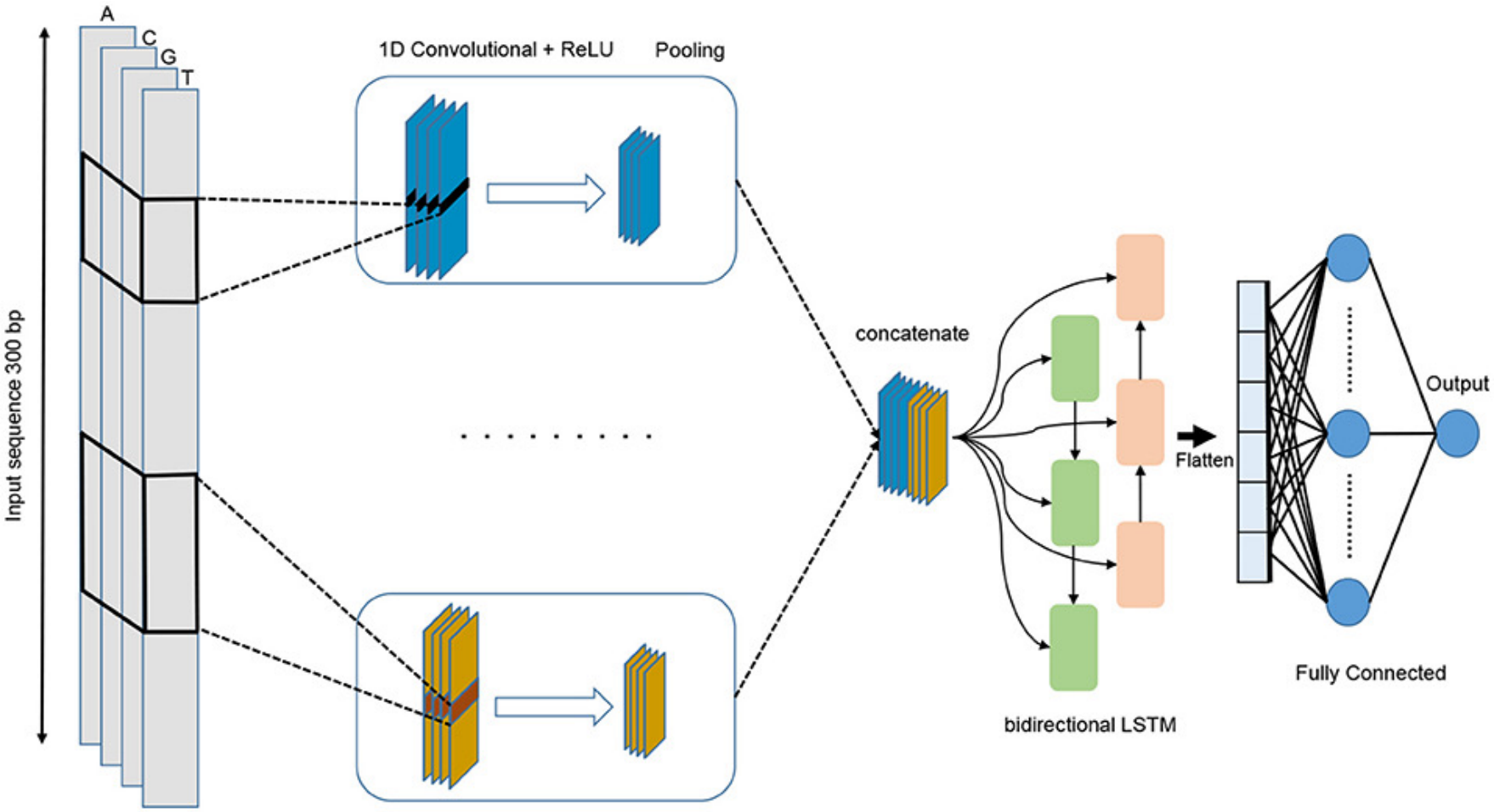


https://cs231n.github.io/convolutional-networks/

Vocabulary

- You still need to choose a vocabulary to turn your text into a number
  - Instead of a count vector you treat each ‘token’ as it’s own class and feed all of them to the network
  - Normally done with something called one-hot encoding
  - Turn each word into a vector with the length of the number of tokens in your vocabulary with one entry = 1 to designate the word and everything else being zero
  - Vocabularies can be bit, so often programs work with a short-hand just labeling the work by an integer.
- Vocabulary can still be words or k-mers, but they also can be characters or 1-mers since all the information is being processed

Example – Deep Promoter



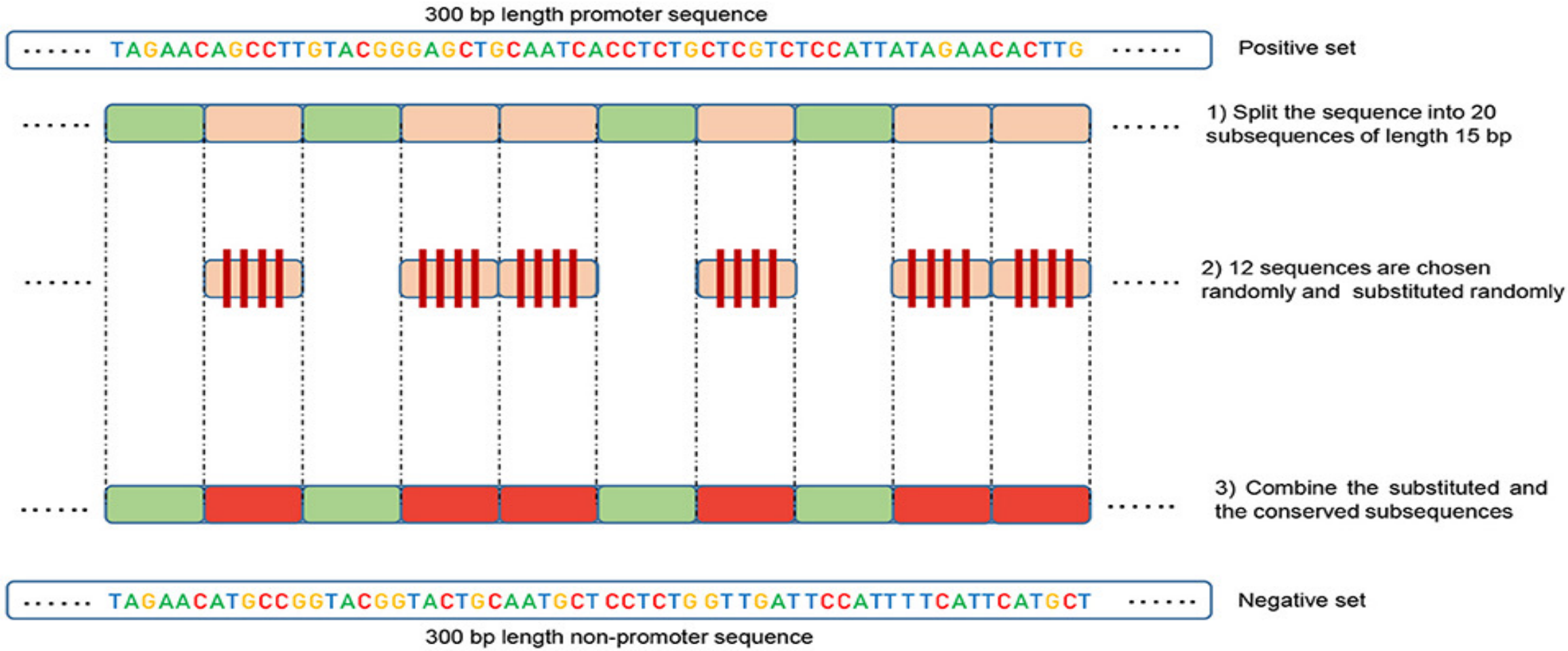

```
def forward(self, x):
    x = x.permute(0, 2, 1)
    x = self.pconv(x)
    x = x.permute(0, 2, 1)
    x = self.bilstm(x)
    x = self.flatten(x)
    x = self.fc(x)
    return x
```

https://github.com/egochao/DeePromoter/blob/main/modules/deepromoter.py

https://doi.org/10.3389/fgene.2019.00286

A promoter, as related to genomics, is a region of DNA upstream of a gene where relevant proteins (such as RNA polymerase and transcription factors) bind to initiate transcription of that gene. – genome.gov

Negative Samples



Workflow of Sequence Data

- 'Lab Magic' is used to identify region of interest in the genome
- Select a window around each region to get 'positive' samples
  - Promoters, ESRRB sites etc.
  - Divide into train and test set
- Create negative samples either from substitutions, permutations, or other sequences
  - Avoid negative sets that can entirely reject with 'trivial' measures like GC content
- Apply to new sequences for prediction
- Use importance measures to identify interesting motifs

A Note on Data

A simple text file with positive sequences is very convenient for the ML community.

main DeePromoter / data / human / TATA / **hs_pos_TATA.txt** Go to file ···

egochao upload data and basic dataset loader Latest commit `cf2c65e` on Jul 6, 2021 **History**

1 contributor

CLICK ME2929 lines (2929 sloc) | 861 KB Raw Blame

```
1  CTCCACTTTTTCTCACGTTTATCTGAGCGAAAACAAGCACGGTTCGGCAGCCTCCTTTCCCAGCCCTACCTTTGTGCTGCAAAAGCGAAAATTCAAAAGCCAAGTACAATAGGAGACCGCCCACCCTGGCTCCCTCGTGACACGA
2  CCAGCAGATGGAAAACAGGACAATGTAACACTGTTCTTATCATCACTATCAGCTGGGACCAGAACAGACACTCAATAAACAGCCTCACACTACAATGAAGCTTGGAGAACAAAGGAGCATCAAAGGGACATGGAGGGCAAGGGTA
3  TCAGAGAAACTGGTCTCTTGATAATAGCCATAGATTACATACTGTGGTCTTCCTCTACATAGACCCTACCTCACCTACCACTCCTGGTCTTAGCTGAAAAACAGGCTAGCCTCGACTCATACTGTCATTTCCTATCCTCCCACTG
4  GAAAAGGTAAAAATACAAGTATGGCTGTTGTTGCTTCAGCCGGGCCTTAATTACCAGACACAAGAAGCAGTTCAGAAATCTGGTCTCTGTTGTTGTAGAGGTCACCAAATCATTAACATCGTCAATCTGGCAGTAGCTAATTTAA
5  CAGGGTCTGCAGAGAGAGAGTGGGGGAGTGCGGGTCGGAGCTTCTGCGCGGACCGGGGCGGGGCACCTCTGGAGGGCAGGGGCCTCTGGTCTCTGGGAGGGGAGGGAATTGACCAATGGGGAGAGAGCCCATATTTGCTCTCAGG
6  AGAATGAGTGCGTCGGGAACCCCAAAACCCTAGATCAAGATACAAGGGGAGTCGACGGCCCGGAAACCTCGCTGCGGGTGCCCTAAGCGCGAGGCCGCCCGGGCAGGGCCGCCCAGGGACCGGGGCGCTGGGTGCGGGGCACGGG
7  ACCAGGGAAGATGAGGCTCCCTTTGCACCCAAAATTTGGGGAGAGGCGAAAGGGGACCGGATGCCCTAGCGGAGCAGAATCCGGCGAAGGGCAGGACCTGAAGGGTTTGGGGAATGCCCTGAATGTGCAGAGCGCGGGGGGGGGG
8  TGCAAAAAGAGCAAAATCTAGGTGAAATGTATTGTTTAAACTTTGATTACCAACCTTGAAAAGGAACATATTAACCAGTGTCTTCTGATAAGCAGATCACTTGCCTCATGTCTTAGAATCCAGTAGGTGGCCCCTTGGCCATGAA
```

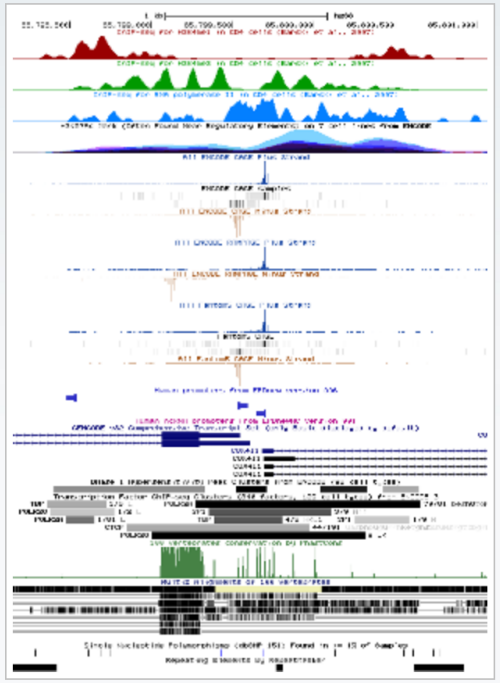
A Note on Data

quality promoters compared to ENSEMBL promoter collection (Dreos et al., 2012) and it is publically accessible at https://epd.epfl.ch//index.php. We downloaded TATA and non-TATA promoter genomic sequences for each organism from EPDnew. This operation resulted in obtaining four promoter datasets namely: Human-TATA, Human-non-TATA, Mouse-TATA, and Mouse-non-TATA. For each of these datasets, a negative set (non-promoter sequences) with the same size of the

Links to databases that don't have your processed data aren't particularly helpful!

hsEPDnew, the *Homo sapiens* (human) curated promoter database

Version	006
Coverage	**29598** promoters **16455** genes
Genome assembly	H. sapiens (Dec 2013 GRCh38/hg38)
Gene annotation	Gencode (v28)
Based on data from	Riken/ENCODE CAGE data downloaded from UCSC FANTOM5 data Rampage data EPD (old)
Documentation & Viewer(s)	Promoter assembly pipeline description EPD viewer - hg38 (track content) EPD viewer - hg19 (track content) GM12878 viewer (hg19) FANTOM5/ZENBU (hg38) DBTSS/KERO (hg38) refTSS (hg38) SwissRegulon (hg19)



Promoters shown above: EMC8 and COX4I1.

Conclusions

- Text analysis techniques can be used on raw sequence data to identify patterns and create predictors
- Things to consider:
  - Structured predictors k-mer counts
    - May not be powerful enough for complicated problems
  - Unstructured predictors deep-learning
    - More overhead in development and computational resources

- **Suggestion:** Always keep your broader goals in mind when evaluating your models