Machine Translation

Alejandro Ciuba
LING1330 – Introduction to Computational Linguistics
December 1st, 2022

History slides based from a presentation by Steve Sloto

Overview

1. Historical Overview
   - ➢ Early models to the modern day
   - ➢ Different approaches – Rules-Based vs. Statistical
   - ➢ Corpora types – Parallel, Monolingual, Dictionaries

2. Important Concepts in Machine Translation
   - ➢ Accuracy measurements
   - ➢ Neural Networks Overview
     1. Back propagation
     2. One-hot vector encoding
     3. Perceptron/FFNN, Encoder-Decoder, RNN/GRU/LSTM, Transformers
   - ➢ Pros & Cons of Neural Network Translations
   - ➢ Recent Research

3. Public-Facing APIs for Machine Learning and Translation
   - ➢ PyTorch & Hugging Face

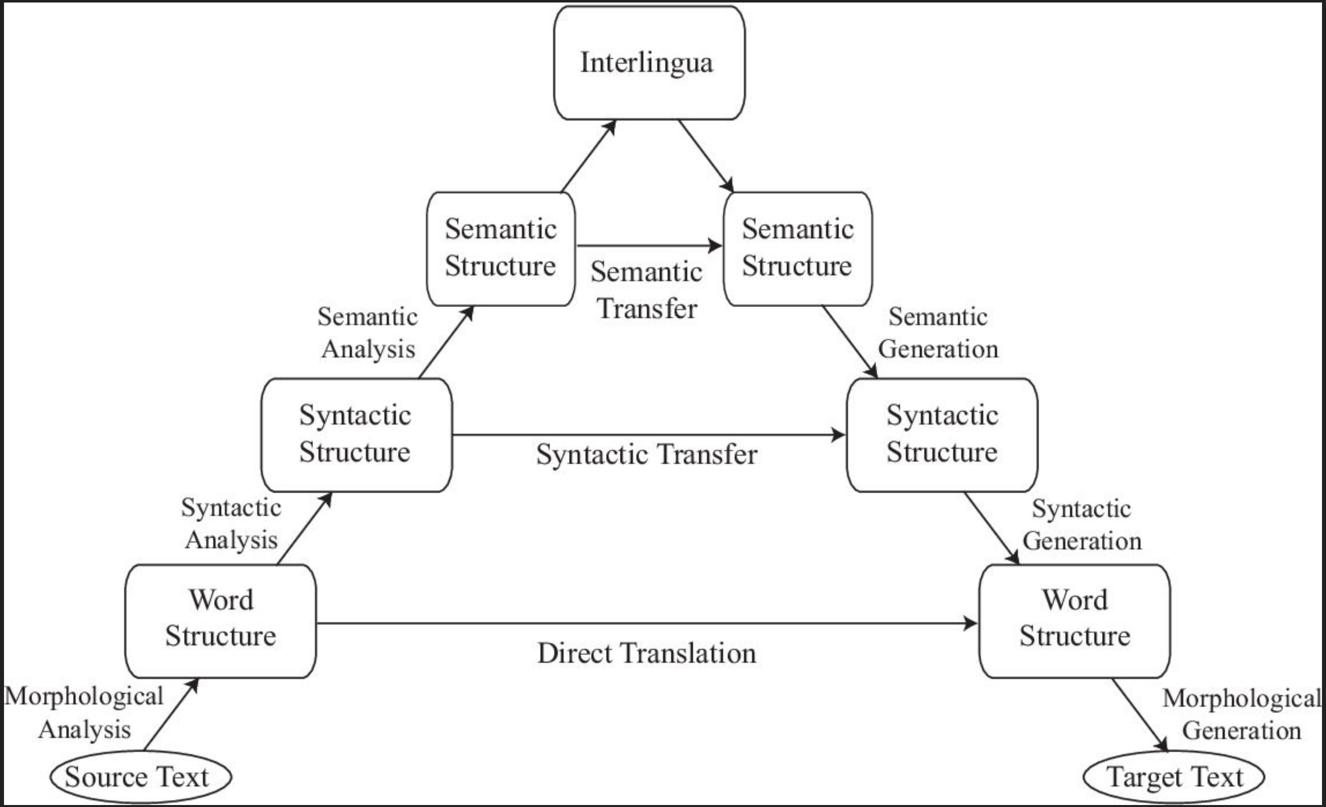
Historical Overview

Historical Overview – The Early Days

- Rule-based Machine Translation
  - Follow a series of deterministic rules which produce a translation
  - Rules were hand-made to each language pair
  - IBM 701 Translator (1954) – Used 6 rules to translate between Russian and English
    1. Assume 1:1 equivalence
    2. Swap words if there is a difference in word order.
    3. Choose target word(s) based on indication in the following source word
    4. Choose target word(s) based on indication in the previous source word
    5. Omit source words that should not appear in the target translation
    6. Add target words that do not appear in the source, but should appear in the target

- Dostert – "Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact.”

Historical Overview – Interlingua



Historical Overview – The Long Reign of Statistics

- Concept: Look at **parallel corpora** to determine what should be translated statistically
  - **Parallel Corpus:** A corpus containing sentence pairs; translations of a sentence
    - Ex. Tamasheq-English-French
  - **Monolingual Corpus:** A corpus containing only sentences/words from one language

- Worked via **The Noisy Channel Model**
  - Find the max probability argmax(P(T|S))
  - You'll learn more about that next week!

Historical Overview – The Rise of a New Approach

- Statistics-based machine translation became the de facto technique for many many years
  - Even as recently as 2015, companies like *Ali Express* were using statistics-based models for translation!

- Neural Networks also had some popularity
  - Recurrent Neural Networks & Encoder-Decoder Models
  - Google Translate became a neural model in November 2016
    - Improved **zero-shot** translation – Translating without an intermediate language

- Something happened however on June 12th, 2017…
  - More on this later!

Important Concepts in Machine Translation

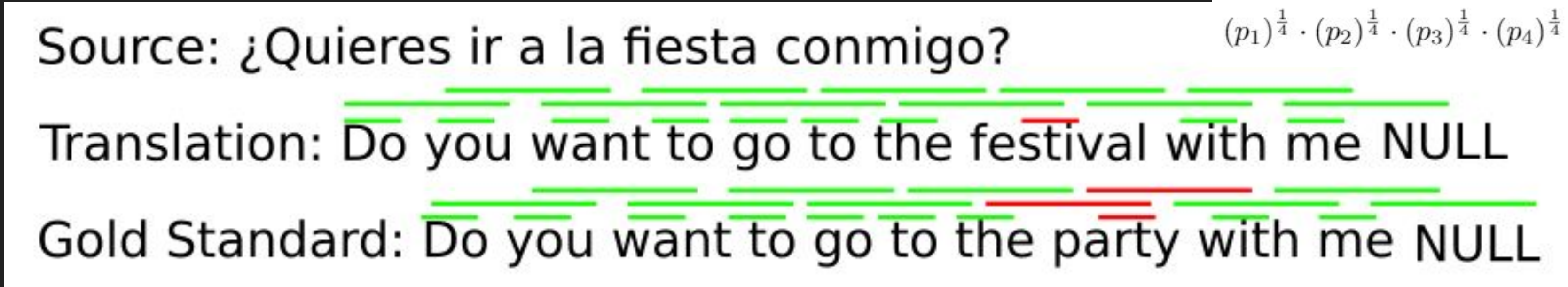
Important Concepts – Basic Measurements

- There are a few ways researchers measure the “accuracy” of a translation
    - Can change between project goals, modality (spoken vs. written), and system-design
- **BLEU (BiLingual Evaluation Understudy):** (The most common measure)
    - Compares n-grams of human-translated gold standard texts to the model’s output
    - The math: $$\log Bleu = \underbrace{\min\left(1-\frac{r}{c},0\right)}_{\text{Brevity Penalty}} + \sum_{n=1}^{4}\frac{\log p_n}{4}$$

Geometric Mean Sum of n-gram precision (1-4)

$$\prod_{n=1}^{N} p_n^{w_n}$$

$$(p_1)^{\frac{1}{4}} \cdot (p_2)^{\frac{1}{4}} \cdot (p_3)^{\frac{1}{4}} \cdot (p_4)^{\frac{1}{4}}$$



Important Concepts – Basic Measurements Cont.

- **BLEU** is good for non-code-switched one-to-one translations, but has drawbacks
  - Based around a human-translated translations
    - Bias in translator style, subject, skill-level, etc.
  - Mostly text-based and unsuitable for code-switched texts

- Word Accuracy/Word Error Rate are more common for voice-based systems
  - Word Accuracy = 1 - WER
  - $$WER = \frac{S+D+I}{N} = \frac{S+D+I}{S+D+C}$$

- New measurements for code-switched data (**Burstiness**), but are not common

Important Concepts – Neural Networks

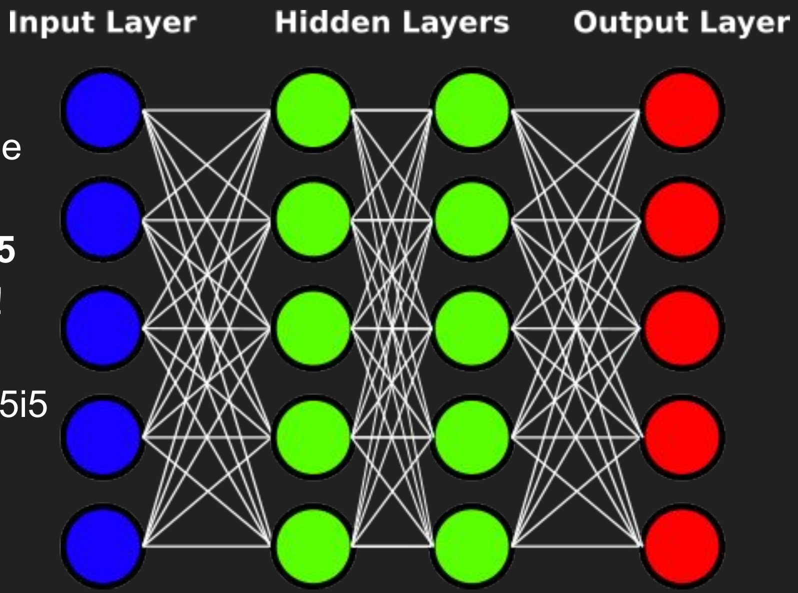
- **THIS IS NOT A COMPREHENSIVE OVERVIEW OF NEURAL NETWORKS!!!**
  - **I suck at them…**

- Important concepts to discuss:
  - Basic Structure
  - Back Propagation
  - One-hot vector encodings
  - Different types of Neural Networks
    - Perceptron/FFNN, Encoder-Decoder, RNN/GRU/LSTM, Transformers

Important Concepts – Basic Structure

- Think of it like a **Graph!!!**
    - A series of nodes (**Outputs**) connected to each other in **Layers**
        - **Edges** are **Weights**
    - Every layer that isn't the actual data itself or the final layer are called **Hidden Layers**

- A Classic Example!!!
    - Each node is connected to every other node
        - **GRAPHS ARE DIRECTIONAL!!!**
    - 5 nodes per layer & 3 layers = 5 * 5 * 3 = **75**
        - This is how many weights there are!!!
    - Equation is just a linear regression
        - Ans = w1i1 + w2i2 + w3i3 + w4i4 + w5i5
            - For all nodes in a layer
            - Ans + **BIAS**



Important Concepts – Back Propagation

- **Back Propagation** is how neural networks learn on data
    - Like regular training data other machine learning models use (think Naive Bayes'!)

- Train the model on batches of data over and over again until we "minimize the loss"
    - **The Loss** is how big the difference between the actual answers from the training data and the answers the model produced are!
    - You get the **Loss** from the **Loss Function** which measures the difference between the results and real answers
        - The **Loss Function** itself changes depending on task, model design, and other factors

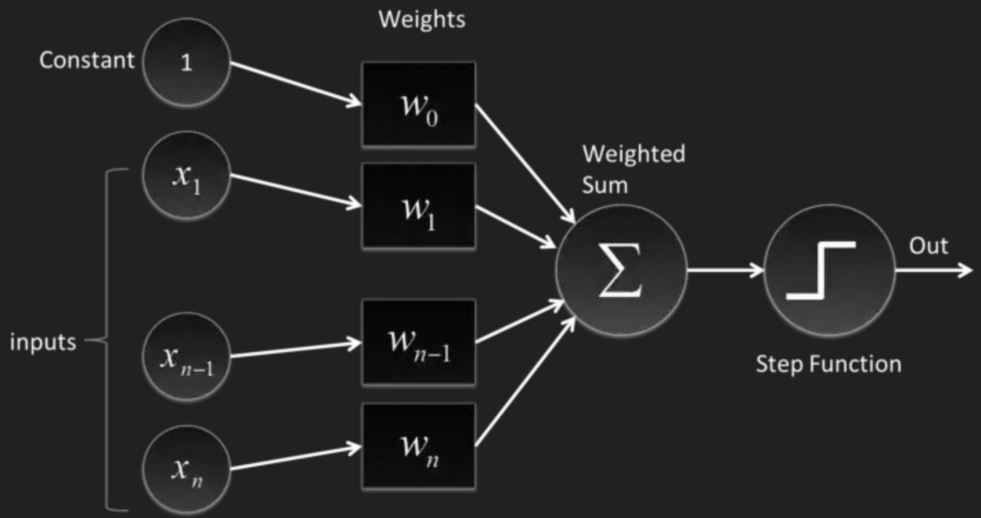
Important Concepts – Encoding Data

- There are many many many ways to encode data for a neural network
  - However, they all involve transforming the data into **vectors** (or matrix… but don't worry)
  - **Vectors** are a series of numbers describing something (they also have direction & magnitude)
    - Ex. Dictionary = {'hello', 'nice', 'meet', 'to', 'world', 'you', '!'}
      Sentence = "hello world!"
      Sentence Vector = {1, 0, 0, 0, 1, 0, 1}
      - This type of encoding is called a **One-Hot Vector** because 1s represent what words are actually in the sentence compared to all the possible words we could have

- Another popular way is just tokenizing the sentence in a specific way
  - Particularly used for Encoder-Decoder models and Transformers
  - Ex. "hello world!" -> ['<s>', 'hello', 'world', '<excl>', '</s>']
    - Changes from model to model; however, a start ('<s>') and end ('</s>') token are popular to include

Important Concepts – Feed Forward Neural Networks

- **Feed Forward Neural Networks** – The first and simplest type of neural model
  - Information is passed from the previous layer to the next directly, going forward through the graph
  - Simplest type **Perceptron**, every node in a layer n is connected once to every node in the previous layer n - 1, with there being only 1 layer in total



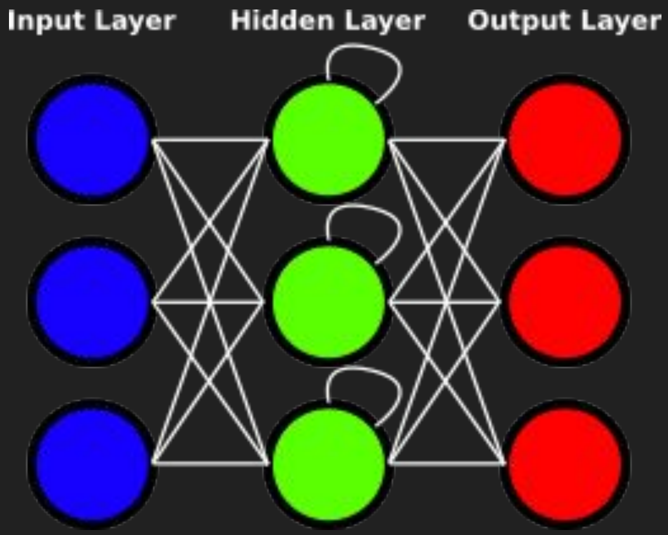
Important Concepts – FFNNs Cont.

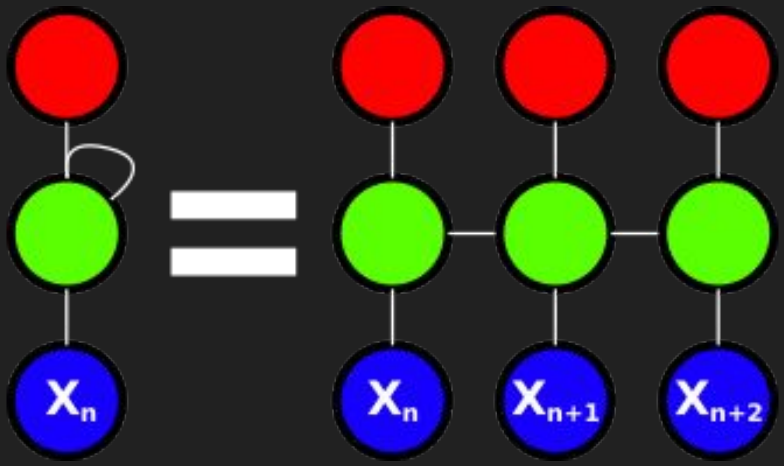
- Advantages:
  - Simple to understand and make
  - Easy to experiment with
  - Simple to train
  - Good learning tool

- Disadvantages:
  - Its simplicity is also a bad thing
    - Terrible for complex tasks
  - Loss of information between layers
    - A given layer n only looks at the previous layer n-1’s information, nothing more, nothing less

Important Concepts – Recurrent Neural Networks

- **<u>R</u>ecurrent <u>N</u>eural <u>N</u>etworks** – Use previous information to affect the output
  - Recursively feed in the input to the result along its journey
    - Outputs from previous steps are included with the current step
  - Features **Directionality** – Can be either uni- or bidirectional





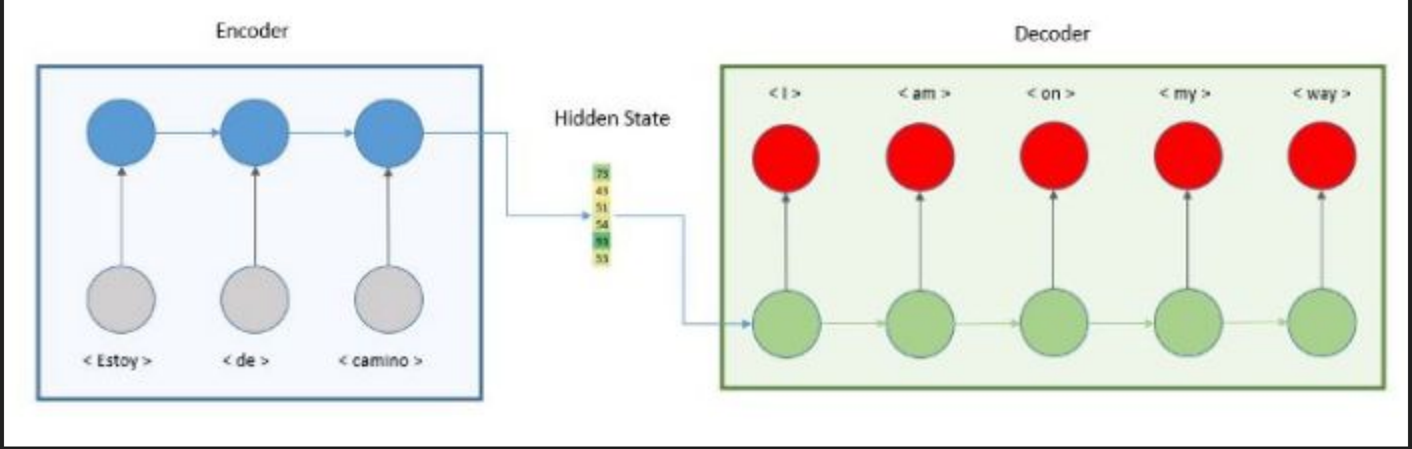
Important Concepts – RNNs Cont.

- Advantages:
  - More connectivity for less
  - “Remembers” certain things
  - Good neural network system to “add-on” to other neural models like convolutional networks

- Disadvantages:
  - The vanishing/exploding gradients
    - The change to each weight either diminishes or increases rapidly
      - We get less improvement over time
      - Or we surpass it!
    - **Long-Short Term Memory** and **Gated Recurrent Units** attempt to solve these issues
      - Add priorities on *what* to remember

Important Concepts – Encoder-Decoder Models

- **Encoder-Decoder Models** – A stack of RNNs produce a hidden state vector which is then fed into another stack of RNNs
  - It *encodes* the data and then *decodes* it later; the hidden state vector contains contextual information

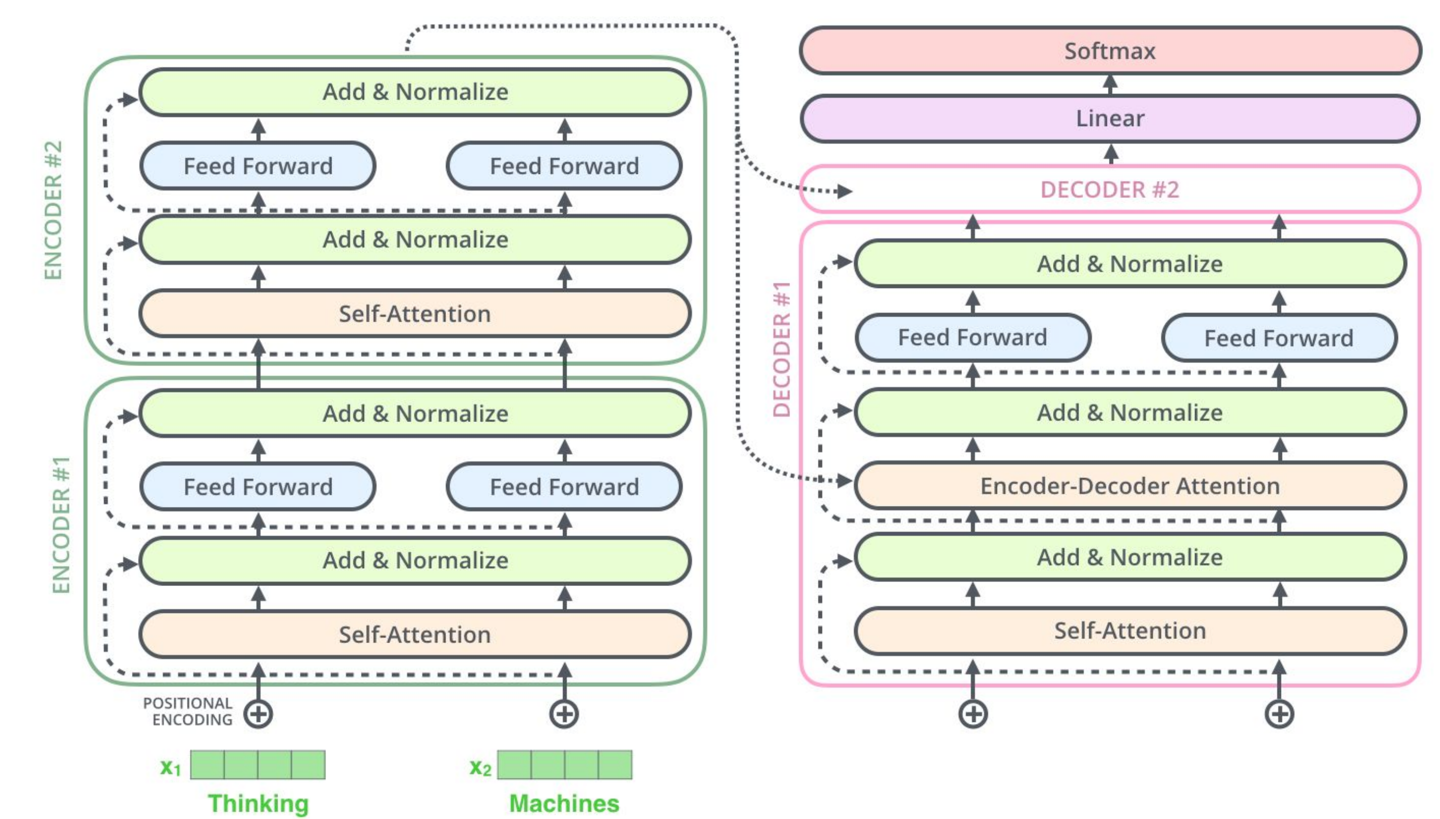


Important Concepts – Transformers

- Remember June 12th, 2017?
  - On that day, several researchers at Google published a very famous paper that changed the field of neural networks and NLP
    - *Attention Is All You Need*

- In this paper, the researchers propose a new neural architecture, called a **Transformer**
  - These are now the de facto standard for many NLP tasks like speech-recognition, question-answer finding, conversation agents, and also… *machine translation!*
  - Focus on this concept called **Attention** a mathematical model for taking in bidirectional context
    - Famous models: **BERT**, **GPT-3**, **T5**, etc.
      - **BERT** in particular has many offshoots: **RoBERTa**, **ALBERT**, **DeBERTa**, etc.

Important Concepts – Transformers Cont.



Important Concepts – Why Neural Networks?

- Transformers are now a de facto approach because they work

- More data = better results*

- They've performed the best compared to statistical and rule-based translation
  - Before transformers, Encoder-Decoder and RNNs were the common models used

- They have a wide-range of uses and can be adapted to many NLP tasks
  - Even in producing vectors for the models! (**BERT** word embeddings)

Important Concepts – Why Not Neural Networks?

- Computationally very **expensive**
  - **Pretrained:** Transformer models can be trained on “general data” for a generalized task
    - NLP tasks, computer-vision tasks, etc.
  - **Fine-Tuning:** Taking these pretrained models and training them more to a specific task
    - Less training than training from scratch with similar results
  - **Hyperparameters:** The weights, biases and other factors (e.g. parameters) inside the model
    - **Random & Grid Search:** Techniques to try to find the best hyperparameters for a task

- The bigger the model, the smaller our understanding
  - Transformers especially are considered “blackboxes”
    - This makes things like bias detection/removal hard

- Lots of data and big models can take weeks to train
  - **Pathways Language Model** (**PaLM**) by Google has 540 BILLION parameters!

Important Concepts – Interesting Developments

- Machine translation for low-resource languages
    - **Low-Resource** just means small amounts of data
- Speech-to-text (STT) systems for code-switched dialogue
    - **Code-Switched:** Speaker(s) switching between languages during a conversation
- Improving the size and scalability of pretrained models
    - Make them smaller, more efficient, and more adaptable to a wider variety of tasks

Public-Facing APIs

Public-Facing APIs – SKLearn, PyTorch & Hugging Face

- **Scikit Learn** is an extremely popular package for learning about and making *machine learning models*
  - Has a wide variety of features and models to use and choose from
  - Despite it being a learning kit, I've seen it used in actual research

- **PyTorch** is a Python package designed to make various *neural network models*
  - More complicated than Scikit Learn, but they can work together
  - Handles the complicated things like the math and the training, but is still extremely flexible
  - Similar packages are Google's **TensorFlow** and **Keras**

- Hugging Face is a website and a group of Python Packages
  - Contain interfaceable pretrained transformer models and various measurement metrics

References & Useful Sources

- https://github.com/AlejandroCiuba/Tamasheq-English-French
- https://www.youtube.com/watch?v=QNBuK9ZNo_0
- https://github.com/AlejandroCiuba/Tamasheq-English-French
- http://ec2-18-219-160-46.us-east-2.compute.amazonaws.com/~narae/vault/ling1330-2021/Guest_Lecture_on_Machine_Translation.pdf
- https://en.wikipedia.org/wiki/L%C3%A9on_Dostert
- https://en.wikipedia.org/wiki/Neural_machine_translation
- https://en.wikipedia.org/wiki/Google_Neural_Machine_Translation#:~:text=In%20November%202016,%20Google%20Neural,,%20in-house%20SMT%20technology.
- https://en.wikipedia.org/wiki/BLEU
- https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b
- https://en.wikipedia.org/wiki/Word_error_rate
- https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53
- https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
- https://en.wikipedia.org/wiki/Feedforward_neural_network
- https://en.wikipedia.org/wiki/Feedforward_neural_network
- https://en.wikipedia.org/wiki/Feedforward_neural_network
- https://www.youtube.com/watch?v=aircAruvnKk&t=915s
- https://www.techtarget.com/searchenterpriseai/definition/recurrent-neural-networks
- https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/
- https://www.ibm.com/cloud/learn/recurrent-neural-networks
- https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
- https://towardsdatascience.com/what-is-an-encoder-decoder-model-86b3d57c5e1a
- https://arxiv.org/abs/1706.03762
- https://jalammar.github.io/illustrated-transformer/
- https://www.topbots.com/leading-nlp-language-models-2020
- https://www.youtube.com/watch?v=ftWlj4FBHTg
- https://towardsdatascience.com/transformers-the-bigger-the-better-19f39f222ee3#:~:text=The%20largest%20model%20so%20far&text=This%20latest%20model%20from%20Google,%2FNVIDIA%20Megatron-Turing%20NLG.
- https://scikit-learn.org/stable/
- https://huggingface.co