

Advanced Text-As-Data - Winter School - Iesp UERJ

Day 3: Introduction to Transformers

Sebastián Vallejo Vera ¹

July 6, 2025

¹ Dept. of Political Science – Western University

Table of Contents

1. The Evolving Methods of NLP
2. Transformers
3. How Do Transformers Work?
4. Final Thoughts

The Evolving Methods of NLP

Evolving Methods

In the last couple of days we have learned about the different goals of NLP: measure some latent characteristic of a set of texts.

- BAG-OF-WORDS: *Wordfish* (Slapin and Proksch, 2008).
- EMBEDDINGS: Party ideological placement, i.e., party-embeddings (Rheault and Cochrane, 2020).
- TRANSFORMERS: Supervised models for text classification (Timoneda and Vallejo Vera, 2025a).

More recently, increased focus on Generative Large Language Models (LLM).

- APPLIED: latent dimensions, i.e., ideology (Kato and Cochrane, 2025; Wu et al., 2023); annotators (Timoneda and Vallejo Vera, 2025b)
- LIMITATIONS: biases from party cues (Vallejo Vera and Driggers, 2025); bias from language (Walker and Timoneda, 2024).
- META: effect of LLMs on respondents.

Evolving Methods

In the last couple of days we have learned about the different goals of NLP: measure some latent characteristic of a set of texts.

- BAG-OF-WORDS: *Wordfish* (Slapin and Proksch, 2008).
- EMBEDDINGS: Party ideological placement, i.e., party-embeddings (Rheault and Cochrane, 2020).
- TRANSFORMERS: Supervised models for text classification (Timoneda and Vallejo Vera, 2025a).

More recently, increased focus on Generative Large Language Models (LLM).

- APPLIED: latent dimensions, i.e., ideology (Kato and Cochrane, 2025; Wu et al., 2023); annotators (Timoneda and Vallejo Vera, 2025b)
- LIMITATIONS: biases from party cues (Vallejo Vera and Driggers, 2025); bias from language (Walker and Timoneda, 2024).
- META: effect of LLMs on respondents.

Evolving Methods

In the last couple of days we have learned about the different goals of NLP: measure some latent characteristic of a set of texts.

- BAG-OF-WORDS: *Wordfish* (Slapin and Proksch, 2008).
- EMBEDDINGS: Party ideological placement, i.e., party-embeddings (Rheault and Cochrane, 2020).
- TRANSFORMERS: Supervised models for text classification (Timoneda and Vallejo Vera, 2025a).

More recently, increased focus on **Generative Large Language Models (LLM)**.

- APPLIED: latent dimensions, i.e., ideology (Kato and Cochrane, 2025; Wu et al., 2023); annotators (Timoneda and Vallejo Vera, 2025b)
- LIMITATIONS: biases from party cues (Vallejo Vera and Driggers, 2025); bias from language (Walker and Timoneda, 2024).
- META: effect of LLMs on respondents.

Transformers

What Are Transformers?

- Transformers are a machine-learning **architecture** used in language-based models.

What Are Transformers?

- Transformers are a machine-learning **architecture** used in language-based models.
- We can think of Transformers as the engines that drive machine-learning models and improve (significantly) their performance.

What Are Transformers?

- Transformers are a machine-learning **architecture** used in language-based models.
- We can think of Transformers as the engines that drive machine-learning models and improve (significantly) their performance.
- Transformers are the architecture behind most (all?) state-of-the-art language models: BERT, RoBERTa, Llama, etc.

Do I Know Any Transformers?

Have I been exposed to this architecture before?



You DO Know Some Transformers

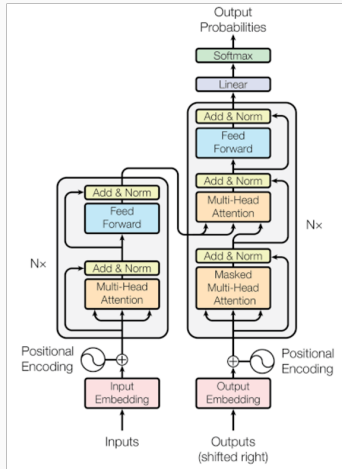
You have!

- Google Translate (and prediction).
- ChatGPT, Grok 🤖, etc.
- GitHub Copilot.
- Automatic caption generation in social media.

GPT = Generative Pre-trained Transformers

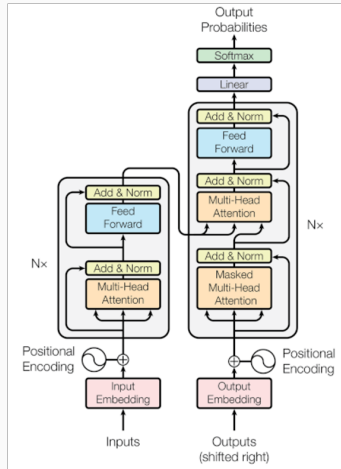
How Do Transformers Work?

The Encoder - Decoder Architecture



- It's complicated...

The Encoder - Decoder Architecture



- It's complicated... but we can break it down into parts.

Encoder Mechanism

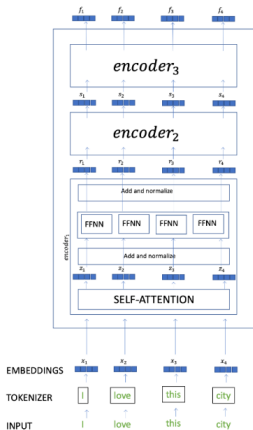


Figure 1: Diagram of a three-stack encoder of a Transformers model. Input text is tokenized and given an initial embedding (vectorized representation) simplified in our figure as x_1 through x_4 . The initial embeddings are transformed as they enter the first *encoder₁*. In it, the self-attention mechanism updates the embeddings (z_1 through z_4), which are then passed through a feed-forward neural network. They exit the encoder as a more accurate set of embeddings (r_1 through r_4). The process is repeated for all encoders in the neural network. For example, pre-trained BERT-base models use 12 encoder layers.

Masking

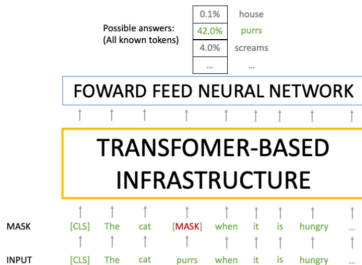
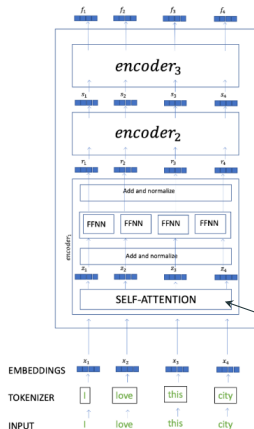


Figure 2: Representation of the Masked Language Modeling. Transformer models randomly mask 15% of tokens. After running the corpus through the Transformers and neural-network architecture, it asks the model to predict the masked word. Transformer models then calculate loss and the required gradient changes to optimize the model's weights and obtain better representations.

Encoder Mechanism



Attention Is All You Need

Figure 1: Diagram of a three-stack encoder of a Transformers model. Input text is tokenized and given an initial embedding (vectorized representation) simplified in our figure as x_1 through x_4 . The initial embeddings are transformed as they enter the first *encoder₁*. In it, the self-attention mechanism updates the embeddings (z_1 through z_4), which are then passed through a feed-forward neural network. They exit the encoder as a more accurate set of embeddings (r_1 through r_4). The process is repeated for all encoders in the neural network. For example, pre-trained BERT-base models use 12 encoder layers.

Self-Attention Mechanism

The self-attention mechanism ('Attention is All you Need') is the revolutionary component of the architecture (Vaswani et al., 2017).

- Each input consists of 'queries' and 'keys' matrices. In the sentence 'Mary likes books', to give meaning to the word 'Mary' (query) we look at the whole sentence, and find the word that is most related to it, in this case, 'likes' (key).

Self-Attention Mechanism

The self-attention mechanism ('Attention is All you Need') is the revolutionary component of the architecture (Vaswani et al., 2017).

- Each input consists of 'queries' and 'keys' matrices. In the sentence 'Mary likes books', to give meaning to the word 'Mary' (query) we look at the whole sentence, and find the word that is most related to it, in this case, 'likes' (key).
- The self-attention mechanism will estimate the distance (dot product) between the vector representation of every query to the already provided vector representation of every key.

Self-Attention Mechanism

The self-attention mechanism ('Attention is All you Need') is the revolutionary component of the architecture (Vaswani et al., 2017).

- Each input consists of 'queries' and 'keys' matrices. In the sentence 'Mary likes books', to give meaning to the word 'Mary' (query) we look at the whole sentence, and find the word that is most related to it, in this case, 'likes' (key).
- The self-attention mechanism will estimate the distance (dot product) between the vector representation of every query to the already provided vector representation of every key.
- Note that, while the tokens can be the same, the representations will be different (as they come from previous layers).

Self-Attention Mechanism

The self-attention mechanism ('Attention is All you Need') is the revolutionary component of the architecture (Vaswani et al., 2017).

- Each input consists of 'queries' and 'keys' matrices. In the sentence 'Mary likes books', to give meaning to the word 'Mary' (query) we look at the whole sentence, and find the word that is most related to it, in this case, 'likes' (key).
- The self-attention mechanism will estimate the distance (dot product) between the vector representation of every query to the already provided vector representation of every key.
- Note that, while the tokens can be the same, the representations will be different (as they come from previous layers).
- Mathematically, we obtain the dot product of K (eys) and Q (ueries) such that $Attention(Q, K, V) = softmax(QK^T)/\sqrt{d_k}V$ where d is the dimension of K and used as a scaling factor.

Neural View of Self-Attention

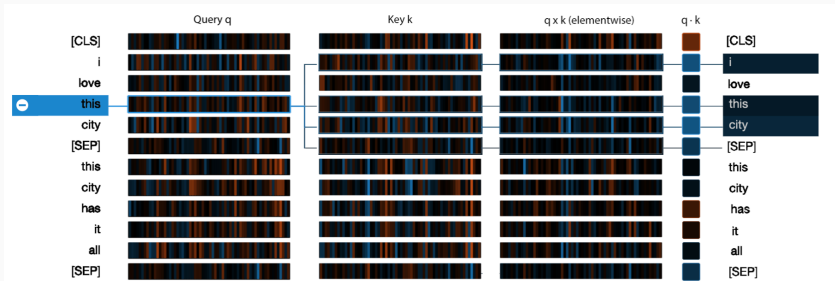


Figure F.2: Neuron view of the attention pattern at Layer 4, Head 1 of the BERT-base pretrained model. The neuron view visualizes the query and key vectors that are used to compute attention. Each color band represents a single neuron value, where color intensity indicates the magnitude and hue the sign (blue=positive, orange=negative).

Positioning Encoding i

- A large corpus cannot be fed in its entirety into the encoding process. Thus, it is broken down into a list of vectors of size 512.
- As the corpus is broken down, key elements of the structure are maintained: [CLS] and [SEP] represent the beginning and the end of a sentence, respectively.
- It also adds a positional encoding to the list of vectors to maintain the order of each token in the sentence. Since embeddings flow independently through the FFNN, position encoding (a vector) helps maintain the order while also providing a relation between words.
- the position encoding (PE) adopts a sinusoidal form:

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{model}})$$

Positioning Encoding ii

- The intuition behind it is that position encoding allows the model to easily find the position of each token in a batch similar. It also allows to count the number of steps between tokens: the farther the distance, the faster it decays (the weaker the relationship between these tokens).

Vaswani et al. (2017) writing about position encoding say that “we chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k , PE_{pos+k} can be represented as a linear function of PE_{pos} ”. While there is no formal proof on their paper of this assertion, it apparently works.

What Does it All Mean? i

This is a lot! It can be a bit overwhelming as well. We will simplify it all when we implement it, but for now the most pressing question should be: what do we do with all of this?

- First, just like you learned with word embeddings, we can **pre-train models**. Well, not 'we'... people* with a lot of computer power. *For example, Google, Meta, and OpenAI.
- Pre-trained models are encoders: they create representations of tokens, *encoding* these with 'meaning'. These are SOTA **understanding** text: BERT, RoBERTa.
- We could also train models to generate text (i.e., GPT or LLM).

What Does it All Mean? ii

- Generative Pre-Trained Transformers (GPT) or Generative Large Language Models (LLM) are decoders: they take some input (e.g., a question in a search query), interpret it (i.e., *decode it*), and produce some response that matches the input. These are SOTA **producing** text (next token prediction): Llama, OpenAI.
- Or we can combine both for sequential tasks, such as translation (e.g., T5 or BART). These tend to be slower and less efficient and for most tasks it is better to use BERT-based models or LLMs.

Final Thoughts

We have learned the basic mechanisms underlying the Transformers infrastructure. Yet, you still do not know what to do with it. Here is what we will do next:

1. Learn about BERT-based pre-trained models.
2. Learn how to implement the aforementioned BERT-based pre-trained models.
3. Learn about Large Language models (LLMs), their applications and limitations.
4. Learn how to implement LLMs.

Thank you!

Questions? Comments? Break?

References

- Kato, Ken and Christopher Cochrane. 2025. "KOKKAI DOC: An LLM-driven framework for scaling parliamentary representatives." *arXiv preprint arXiv:2505.07118* .
- Rheault, Ludovic and Christopher Cochrane. 2020. "Word embeddings for the analysis of ideological placement in parliamentary corpora." *Political analysis* 28(1):112–133.
- Slapin, Jonathan B and Sven-Oliver Proksch. 2008. "A scaling model for estimating time-series party positions from texts." *American Journal of Political Science* 52(3):705–722.

- Timoneda, Joan C and Sebastián Vallejo Vera. 2025a. “BERT, RoBERTa or DeBERTa? Comparing Performance Across Transformers Models in Political Science Text.” *The Journal of Politics* 87(1):000–000.
- Timoneda, Joan C and Sebastián Vallejo Vera. 2025b. “Memory Is All You Need: Testing How Model Memory Affects LLM Performance in Annotation Tasks.” *arXiv preprint arXiv:2503.04874* .
- Vallejo Vera, Sebastián and Hunter Driggers. 2025. “Bias in LLMs as Annotators: The Effect of Party Cues on Labelling Decision by Large Language Models.” *arXiv preprint arXiv:2408.15895* .
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser and Illia Polosukhin. 2017. “Attention is all you need.” *Advances in neural information processing systems* 30.

Walker, Christina and Joan C Timoneda. 2024. “Identifying the sources of ideological bias in GPT models through linguistic variation in output.” *arXiv preprint arXiv:2409.06043* .

Wu, Patrick Y, Jonathan Nagler, Joshua A Tucker and Solomon Messing. 2023. “Large language models can be used to estimate the latent positions of politicians.” *arXiv preprint arXiv:2303.12057* .