WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

A Comparison of Transformer and Autoregressive LLM Designs

Saravanan Hari Baskaran

Vellore Institute of Technology
Email: hsaravanan29@gmail.com

ABSTRACT

Language models play a pivotal role in natural language processing, and two prominent architectures, autoregressive and Transformer-based models, have emerged as leading contenders in this domain. This comparative analysis seeks to provide a comprehensive overview of the characteristics, strengths, and limitations of both approaches.

Autoregressive language models, exemplified by the GPT (Generative Pre-trained Transformer) series, generate text sequentially, conditioning each token on its predecessors. This sequential generation fosters coherent text but hampers inference speed. On the other hand, Transformer-based models, including BERT, RoBERTa, and variants, leverage self-attention mechanisms for parallel processing during training and inference, resulting in faster computation. However, they may produce less contextually relevant text when it comes to text generation tasks.

This analysis explores the use cases, advantages, and disadvantages of autoregressive and Transformer language models. It underscores that the choice between the two depends on the specific task, computational resources, and the need for a balance between coherence and speed. Some tasks benefit from the sequential context provided by autoregressive models, while others require the parallelization offered by Transformers for real-time applications.

The comparative analysis further discusses how large-scale models like GPT-3 and T5 have propelled the field of natural language processing, along with the significance of transfer learning through pre-training and fine-tuning in Transformer models.

In conclusion, this comparative analysis sheds light on the pivotal role played by autoregressive and Transformer-based language models in NLP, offering insights for researchers and practitioners in choosing the most suitable architecture for their specific applications, thus contributing to the ongoing evolution of language modeling in the field.

Keywords: LLM Architecture, BERT, NLP, GPT, Language Model

1. Introduction

Autoregressive and Transformer-based language model architectures represent two distinct paradigms for natural language processing. Understanding the differences between these approaches can provide valuable insights into their strengths and limitations.

Autoregressive and Transformer-based Language Model (LLM) architectures represent two prominent approaches in the field of natural language processing (NLP). Each approach has its own unique characteristics and advantages. In this comparative analysis, we'll provide an introduction to both Autoregressive and Transformer-based LLM architectures, highlighting their key features and differences.

In a comparative analysis, Transformer-based LLMs often outperform autoregressive models in various NLP tasks. Their ability to capture long-range dependencies, parallelize processing, and leverage pre-training makes them a preferred choice for many applications. However, autoregressive models still have their place, especially in cases where sequential generation is essential, or for certain niche applications.

The choice between these architectures depends on the specific requirements of the NLP task and the trade-offs between efficiency, accuracy, and modeling capabilities. Researchers and practitioners continue to explore the strengths and weaknesses of both approaches, with ongoing efforts to develop hybrid models that combine the advantages of both autoregressive and Transformer-based architectures.

Autoregressive Language Models:

1. Concept: Autoregressive models generate sequences of tokens one at a time, conditioning each token on the previously generated ones.
2. Operation: Models like GPT (Generative Pre-trained Transformer) use autoregressive approaches, where the next word in a sequence is predicted based on the previous words. They are trained to maximize the likelihood of the next word given the preceding context.

3. Strengths: These models are proficient in generating coherent, contextually appropriate text and are well-suited for tasks that require sequential generation, such as language translation and text completion.
4. Limitations: Autoregressive models tend to be slower in generating outputs due to their sequential nature, making them less efficient for tasks that demand real-time or high-throughput processing.

Transformer Language Models:

1. Concept: Transformer-based models, like BERT (Bidirectional Encoder Representations from Transformers), leverage attention mechanisms to process the entire sequence simultaneously, allowing for parallelization and more efficient training.
2. Operation: These models can be fine-tuned for a variety of tasks, including language understanding, translation, and sentiment analysis, by employing attention mechanisms that enable capturing context from both directions without explicitly enforcing an autoregressive generation process.
3. Strengths: Transformer models excel in capturing bidirectional context and have demonstrated superior performance on various natural language understanding tasks. They are highly efficient for tasks that require parallel processing due to their ability to process input sequences all at once.
4. Limitations: Transformers may not generate text as fluently as autoregressive models since they do not explicitly model the sequential nature of language. They might also struggle with tasks that necessitate sequential generation or long-range dependencies.

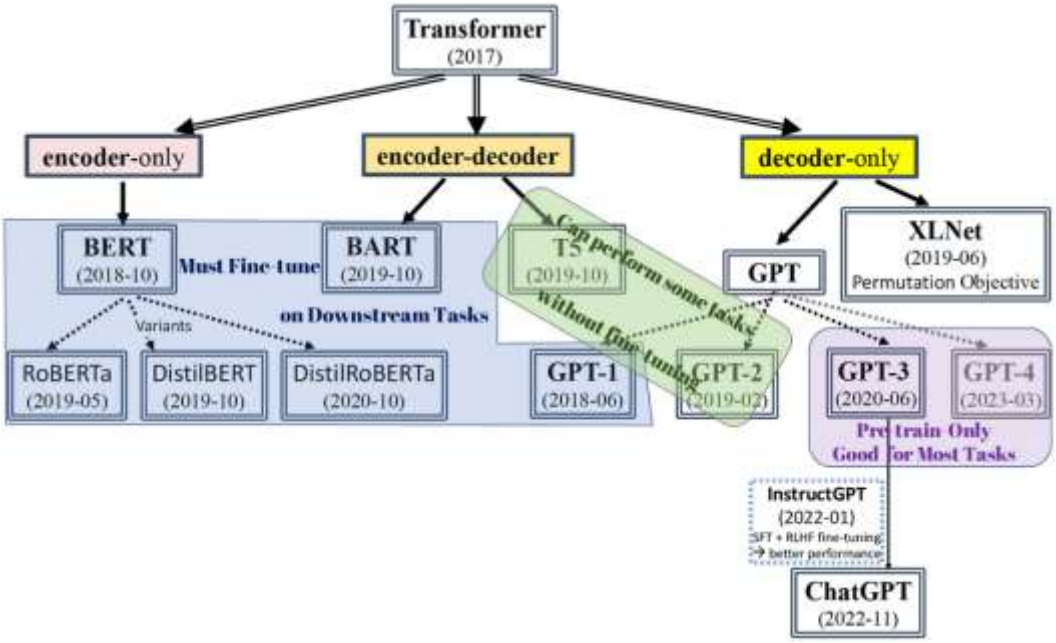

Figure 1: An In-Depth Look at the Transformer Based Models

Comparative Analysis:

1. Autoregressive models perform better in tasks that involve sequential generation, such as text generation, story completion, and dialogue generation.
2. Transformer models are more suitable for tasks that require bidirectional context understanding, such as question-answering, sentiment analysis, and named entity recognition.
3. Autoregressive models are slower in generating outputs, while Transformer models can process inputs in parallel, making them faster and more efficient for tasks that demand high throughput.
4. Researchers are continuously exploring ways to combine the strengths of both architectures to create more powerful and efficient models that can handle a wider range of natural language processing tasks.

2. Inferences Strained

2.1. Bert

BERT, or Bidirectional Encoder Representations from Transformers, is a groundbreaking natural language processing (NLP) model that has revolutionized the field. Introduced by researchers at Google in 2018, BERT is a pre-trained language model that has achieved remarkable success in a wide range of NLP tasks. Unlike earlier models that relied on unidirectional context, BERT uses bidirectional context to understand the nuances and dependencies within a text.

BERT's key innovation lies in its ability to capture contextual information by considering both the left and right context of a word simultaneously. It does this through a deep Transformer architecture, a neural network architecture known for its effectiveness in modeling sequential data. BERT is pre-trained on large corpora of text data, which enables it to learn a rich representation of language and context.

This pre-training stage allows BERT to understand language at a deep level, including word meanings, syntactic structures, and even some aspects of world knowledge. Once pre-trained, BERT can be fine-tuned on specific NLP tasks, such as sentiment analysis, text classification, question answering, and more. Fine-tuning adapts the model to perform a particular task while leveraging the general language understanding it has acquired during pre-training.

BERT's impact on NLP has been substantial, leading to significant improvements in the accuracy and performance of various NLP applications. Its open-source nature has also led to the development of various BERT-based models and derivatives, making it a foundational building block in the world of natural language understanding and processing. The versatility and effectiveness of BERT have firmly established it as a cornerstone in modern NLP research and applications.

2.2. Transformer-XL

Transformer-XL is an advanced variant of the Transformer architecture, specifically designed to address the challenge of modeling long-range dependencies in sequential data, such as natural language text or time series. This model was introduced to overcome the limitations of traditional Transformers when it comes to handling sequences that are much longer than the maximum context window they were originally designed for. Transformer-XL employs a novel approach to capture extensive context information without sacrificing computational efficiency.

"Transformer-XL is a groundbreaking extension of the Transformer architecture, aimed at enhancing the modeling of long-range dependencies in sequential data. Unlike standard Transformers, which rely on a fixed context window, Transformer-XL introduces a novel approach known as 'recurrent positional embeddings' and 'segment-level recurrence.' This innovation allows the model to effectively capture context information from much longer sequences, making it well-suited for tasks that require extensive contextual understanding, such as language modeling and text generation.

2.3. XLNet

The primary innovation of XLNet is its ability to model text bidirectionally (both left-to-right and right-to-left) while still maintaining the advantages of the autoregressive training approach. In essence, it combines the strengths of both autoregressive models like GPT and bidirectional models like BERT. XLNet is a powerful and versatile pre-trained language model that excels at capturing complex contextual relationships in natural language text. Its bidirectional context and permutation-based training make it a valuable tool for a wide range of NLP applications, offering impressive accuracy and generalization capabilities. Researchers and practitioners have continued to build upon XLNet's foundation to develop even more advanced language models in the field of NLP.

2.4. RoBERTa

RoBERTa, a variant of the BERT (Bidirectional Encoder Representations from Transformers) model, is a pre-trained language model designed for various natural language processing tasks. Introduced in 2019, RoBERTa extends BERT's architecture by fine-tuning hyperparameters, training data, and optimization techniques. It focuses on addressing some of the limitations of BERT, including better generalization, understanding of context, and handling of longer sequences. RoBERTa leverages a transformer-based architecture, utilizing multi-layer bidirectional encoders to capture contextual information from input text. By pre-training on massive text corpora and employing advanced techniques like dynamic masking and large-batch training, RoBERTa achieves state-of-the-art performance across a wide range of NLP benchmarks, including tasks such as sentiment analysis, question answering, and natural language inference.

2.5. ALBERT

ALBERT, which stands for "A Lite BERT," is a state-of-the-art natural language processing (NLP) model introduced by Google Research in 2019. It represents an innovative approach to building more efficient and effective NLP models by addressing some of the limitations of the original BERT (Bidirectional Encoder Representations from Transformers) model. The key idea behind ALBERT is model size reduction and parameter efficiency. ALBERT achieves this by sharing parameters across layers and implementing a factorized embedding parameterization. This reduces the number of parameters in the model, making it more computationally efficient while maintaining or even surpassing the performance of larger models.

2.6. T5

T5, which stands for Text-to-Text Transfer Transformer, is a versatile and powerful language model developed by Google's AI research team. It represents a significant advancement in the field of natural language processing (NLP) and has demonstrated state-of-the-art performance across various NLP tasks. T5 is built upon the Transformer architecture and is known for its ability to perform a wide range of text-based tasks using a unified text-to-text framework. This approach enables it to handle different NLP tasks by framing them as text-to-text problems, where the input and output are both text sequences. The key components and features of T5 include its multi-layer Transformer architecture, which allows it to capture complex relationships within the input text, and its extensive pre-training on large-scale datasets, enabling it to learn rich representations of textual data. T5 also utilizes transfer learning techniques, leveraging its pre-trained knowledge to adapt to specific downstream tasks with minimal task-specific fine-tuning. Its effectiveness in handling diverse NLP tasks has made T5 a significant milestone in the development of robust and adaptable language models that can efficiently handle various text-based applications.

2.7. PEGASUS

PEGASUS, which stands for "Pre-trained Global-to-Local Aggregation for Summarization," is a state-of-the-art abstractive text summarization model in the field of natural language processing (NLP). Developed by researchers at Google Research, PEGASUS has garnered significant attention and acclaim for its ability to generate coherent and contextually relevant summaries of long and complex documents. At its core, PEGASUS is based on the Transformer architecture, a neural network architecture known for its excellence in sequence-to-sequence tasks. What sets PEGASUS apart is its innovative pre-training and fine-tuning strategies. The model is pre-trained on a large corpus of diverse text from the internet, which enables it to acquire a comprehensive understanding of language and its nuances.

During pre-training, PEGASUS is trained to perform a document-to-summary task. This means it learns how to take a document, which can be a lengthy piece of text, and produce a concise and coherent summary that captures the essential information and meaning of the document. PEGASUS excels at compressing information while maintaining readability and coherence in its generated summaries.

2.8. ELECTRA

ELECTRA, short for "Efficiently Learning an Encoder that Classifies Token Replacements Accurately," is a highly innovative and efficient pre-training approach for natural language processing (NLP) tasks. Developed by researchers at Google Research, ELECTRA introduces a novel paradigm for training language models that has gained significant attention within the NLP community. ELECTRA represents an innovative approach to pre-training language models for NLP tasks, focusing on token replacement and a discriminator-generator framework. This methodology offers the advantage of improved computational efficiency while maintaining or even surpassing the state-of-the-art performance of existing models. ELECTRA has become a pivotal development in the field of NLP and has opened the door to more efficient and effective pre-training techniques.

2.9. GPT-3

It represents a significant milestone in natural language processing and artificial intelligence. GPT-3 is built on the Transformer architecture, which has revolutionized various NLP tasks. It is characterized by its immense scale, with 175 billion parameters, enabling it to understand and generate human-like text across multiple languages.

GPT-3's core innovation lies in its pre-training process. It is trained on a massive corpus of text data from the internet, allowing it to learn language patterns, grammar, and a wide range of factual knowledge. This pre-training stage equips GPT-3 with a remarkable capacity to generate coherent and contextually relevant text.

3. Comparative Analysis

3.1. Autoregressive Models (AR)

Autoregressive models are a class of sequential models used in various fields, including natural language processing (NLP), time series analysis, and speech recognition. These models are characterized by their ability to generate or predict a sequence of data points, such as words in a sentence, one step at a time. Autoregressive models are particularly common in NLP and are used for tasks like text generation, machine translation, and speech recognition. Here are some key characteristics of autoregressive models:

1. Autoregressive models generate data points one at a time in a sequential manner. For example, in the context of NLP, when generating text, they predict the next word or token based on the preceding words or tokens.
2. Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), are commonly used as the architecture for autoregressive models. These networks maintain recurrent connections, which allow them to store and propagate information over time, making them suitable for capturing dependencies within sequential data.

3. Autoregressive models often rely on conditional probability distributions to make predictions. Given the context of the previously generated data points, they estimate the likelihood of possible next data points and choose the one with the highest probability.
4. During training, autoregressive models typically use a technique called "teacher forcing." This means that they are provided with the actual previous data points as input to predict the next one. This can help improve training stability but may lead to exposure bias, where the model struggles to generate accurate predictions during inference.
5. Autoregressive models can handle variable-length sequences, which is a useful feature in tasks like text generation, where the length of the generated text may vary.
6. Autoregressive models have limitations in capturing long-range dependencies in data, and they can be slow during both training and inference, as they need to process each data point in sequence. This sequential nature can hinder parallelism.

3.2. Autoencoding Models (AE)

Autoencoding Models (AE), specifically Autoencoders, are a class of artificial neural networks used for unsupervised learning and data compression. They have a wide range of applications, including data denoising, dimensionality reduction, and feature learning. Autoencoders work by learning a compact representation (encoding) of the input data and then attempting to reconstruct the input from this representation. The primary goal is to capture the most important features of the input data in a lower-dimensional space.

Here's an overview of the key components and concepts related to Autoencoding Models:

1. The encoder is the first part of the autoencoder. It takes the input data and transforms it into a lower-dimensional representation. This lower-dimensional representation, often called the "latent space" or "encoding," is designed to capture the most important features of the input data.
2. The decoder is the second part of the autoencoder. It takes the encoding produced by the encoder and attempts to reconstruct the original input data from this representation. The quality of reconstruction is a measure of how well the autoencoder has learned to capture the essential information in the data.
3. The loss function (also known as the reconstruction loss) quantifies the difference between the original input and the reconstructed output. Common loss functions for autoencoders include Mean Squared Error (MSE) for continuous data or Cross-Entropy Loss for binary data.
4. The latent space is the lower-dimensional representation learned by the autoencoder. This space typically has a much lower dimensionality than the input data and can be considered a compressed version of the data.
5. In the autoencoder's architecture, the bottleneck layer refers to the layer in the encoder that has the smallest number of neurons. It is responsible for reducing the dimensionality of the data.
6. Autoencoders are trained by minimizing the reconstruction loss using backpropagation and optimization algorithms (e.g., gradient descent). During training, the encoder learns to map the input data to the latent space, and the decoder learns to map the latent space back to the input space.
7. There are various types of autoencoders, including the standard feedforward autoencoder, convolutional autoencoder (used for image data), recurrent autoencoder (used for sequential data), denoising autoencoder (trained to remove noise from data), and variational autoencoder (VAE), which incorporates probabilistic modeling for generative purposes.

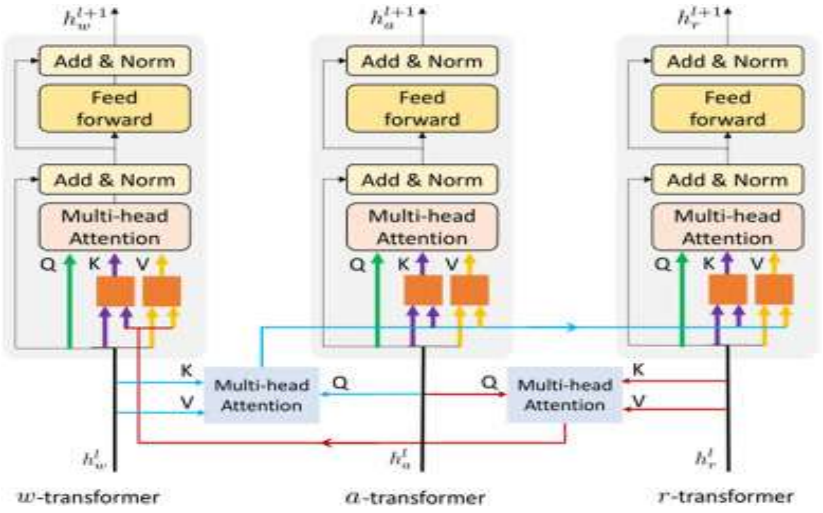

Figure 2: Survey: Transformer based video-language pre-training

Autoencoders have several practical applications, including:

1. Autoencoders can be used to compress data into a more compact representation, which is particularly valuable for reducing storage and transmission requirements.
2. By learning the normal patterns in the data, autoencoders can be used to detect anomalies or outliers.
3. Autoencoders can help reduce the dimensionality of high-dimensional data while retaining its essential features.
4. Autoencoders can be used to learn useful features from raw data, which can be beneficial in downstream supervised learning tasks.
5. Denoising autoencoders are useful for removing noise from images or other types of data.
6. Autoencoding models, and especially their variations like variational autoencoders (VAEs), have become fundamental in modern machine learning and deep learning due to their versatility and ability to capture meaningful representations from complex data.

4. Related Work

With an emphasis on major findings, this section examines recent publications and research articles that have significantly advanced the field of music research. A system for providing real-time musical accompaniment was proposed by the authors of. It used a computer-driven orchestra that learned from and followed a soloist through the use of hidden Markov models and Kalman filter-like models. The authors of provide a thorough analysis of deep learning methods for handling audio inputs, including speech, music, and ambient sound processing.

Other noteworthy contributions to the subject of music study include the identification of writings in the Hathi Trust Digital Library that belong to the Black Fantastic genre through the application of machine learning and word feature analysis. Another study looked at deep learning techniques for estimating musical composition difficulty, leading to automated difficulty-controlled music creation.

In order to reduce the costs involved in producing unique music for promotional videos, the writers also covered the use of artificial intelligence in music production. Combining local correlations with global self-attention, the Orthogonal Transformer has become a well-known vision transformer foundation in computer vision. Pure transformer models have demonstrated outstanding results in jobs involving video categorization. The possible effects of cutting-edge technology on the music business were highlighted by examining the relationship between quantum computing and song production. Within the field of music harmonization, a novel approach was presented for creating fresh harmonization of jazz classics by the fusion of chords and melodies from other tunes.

5. Methodology

Autoregressive and Transformer-based Language Model (LLM) architectures are two popular approaches for natural language processing tasks, such as text generation, machine translation, and text classification. Here's an overview of the methodology for each of these architectures:

1. Autoregressive Language Model:

1) Autoregressive language models are typically based on recurrent neural networks (RNNs) or more advanced models like LSTMs (Long Short-Term Memory) and GRUs (Gated Recurrent Units).
2) The core idea behind autoregressive models is that they generate text one token at a time, with each token being conditioned on the previous ones.
3) The training process involves minimizing the negative log-likelihood of the next token given the previous context. The loss is typically computed using cross-entropy.
4) During training, the model is exposed to a large dataset of text and learns to predict the probability distribution of the next token given the preceding context.
5) At inference time, the model generates text by sampling from this learned probability distribution or by using a decoding strategy like beam search.

2. Transformer-based Language Model:

1) Transformer-based language models are built on the Transformer architecture, which was introduced in the paper "Attention Is All You Need" by Vaswani et al.
2) The core innovation of the Transformer is the self-attention mechanism, which allows the model to capture long-range dependencies in the input sequence.
3) The training methodology for Transformer-based LLMs involves a large-scale, unsupervised pretraining stage followed by fine-tuning on specific downstream tasks (e.g., text classification, machine translation).

4) During pretraining, the model is trained on a massive corpus of text to learn a strong language representation.

5) Fine-tuning involves training the pretrained model on a smaller labeled dataset for a specific task. This stage adapts the pretrained model to perform well on the target task.

6) The loss function used during pretraining is typically a combination of language modeling objectives, such as autoregressive language modeling and denoising autoencoding, which helps the model learn useful language representations.

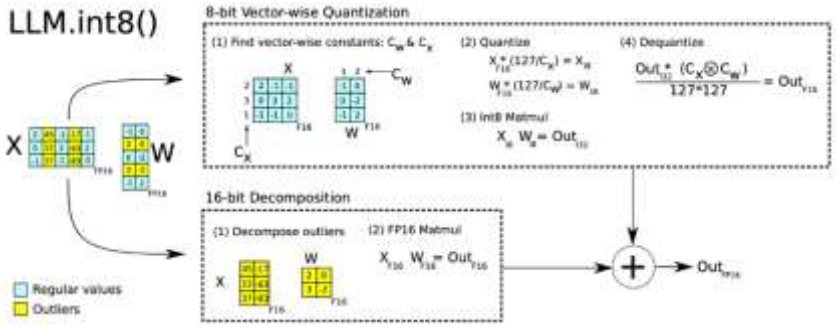

Figure 3: Large Transformer Model Inference Optimization

Overall, both autoregressive and Transformer-based language models have their strengths and weaknesses. Autoregressive models like LSTMs are sequential and can be computationally expensive, but they have been the go-to choice for many years. Transformer-based models, on the other hand, have shown remarkable performance improvements and can handle longer-range dependencies more efficiently. They have become the dominant architecture in the field of NLP, with models like BERT, GPT, and others setting new benchmarks for various NLP tasks. The methodology for training and using these models continues to evolve with ongoing research in the field.

6. Results

Autoregressive and Transformer-based Language Model (LLM) architectures represent two distinct approaches to natural language processing, each with its own set of advantages and limitations. Here, I'll present a brief conclusion on both architectures:

1. Autoregressive Language Models:

Autoregressive language models, such as GPT (Generative Pre-trained Transformer) series, have been at the forefront of natural language processing due to their impressive performance in various NLP tasks, including text generation, translation, and sentiment analysis.

These models are based on Recurrent Neural Networks (RNNs) and are trained to generate text one word at a time, conditioning on the previously generated words. This sequential nature makes them computationally expensive and slow to train and generate text.

Autoregressive models tend to suffer from issues related to generating coherent and contextually relevant text, as they often rely on a fixed-length context window, which can lead to issues like repetition and inconsistency.

Despite their limitations, autoregressive models have been instrumental in demonstrating the power of pre-trained language models, leading to advancements in a wide range of NLP applications.

2. Transformer-based Language Models:

Transformer-based language models, such as BERT (Bidirectional Encoder Representations from Transformers) and its variants, revolutionized the field of NLP by introducing a novel architecture that doesn't rely on sequential processing.

These models use self-attention mechanisms to capture context from both left and right sides of a word, enabling bidirectional understanding of the input text. This leads to improved context modeling and better representations for various NLP tasks.

Transformers have been widely adopted for various NLP tasks, including question answering, text classification, and named entity recognition. They have also paved the way for transfer learning, where pre-trained models are fine-tuned for specific tasks.

The parallelization and efficiency of the Transformer architecture make it faster and more scalable compared to autoregressive models, leading to practical advantages in large-scale applications.

7. Conclusions and unborn Work

In conclusion, both autoregressive and Transformer-based LLM architectures have made significant contributions to natural language processing. Autoregressive models excel in their ability to generate creative and contextually coherent text, while Transformer-based models have ushered in a new era of NLP by offering better contextual understanding and scalability. The choice between these architectures depends on the specific NLP task at hand and the trade-offs between model size, training time, and performance. Researchers and practitioners continue to explore and refine these architectures to push the boundaries of NLP capabilities.

References

[1]. Humphrey, E.J.; Bello, J.P. Rethinking Automatic Chord Recognition with Convolutional Neural Networks. In Proceedings of the 11th IEEE International Conference on Machine Learning and Applications (ICMLA), Boca Raton, FL, USA, 12–15 December 2012; pp. 357–362.

[2]. Mauch, M.; Dixon, S. Approximate Note Transcription for the Improved Identification of Difficult Chords. In Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR), Utrecht, The Netherlands, 9–13 August 2010; pp. 135–140.

[3]. Temperley, D. The Cognition of Basic Musical Structures; MIT Press: Cambridge, MA, USA, 2004.

[4]. Krumhansl, C.L.; Kessler, E.J. Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys. Psychol. Rev. 1982, 89, 334.

[5]. Faraldo, Á.; Gómez, E.; Jordà, S.; Herrera, P. Key Estimation in Electronic Dance Music. In Advances in Information Retrieval, Proceedings of the 38th European Conference on IR Research (ECIR), Padua, Italy, 20–23 March 2016; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9626, pp. 335–347.

[6]. Noland, K.; Sandler, M. Signal Processing Parameters for Tonality Estimation. In Proceedings of the Audio Engineering Society Convention 122, Vienna, Austria, 5–8 May 2007. Pauws, S. Musical Key Extraction from Audio. In Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR), Barcelona, Spain, 10–14 October 2004. Temperley, D. WWhat's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered. Music Percept. 1999, 17, 65–100.

[7]. Giorgi, B.D.; Zanoni, M.; Sarti, A.; Tubaro, S. Automatic Chord Recognition based on the Probabilistic Modeling of Diatonic Modal Harmony. In Proceedings of the 8th International Workshop on Multidimensional Systems, Erlangen, Germany, 9–11 September 2013; pp. 1–6.

[8]. Mauch, M.; Dixon, S. Simultaneous Estimation of Chords and Musical Context From Audio. IEEE Trans. Audio Speech Lang. Process. 2010, 18, 1280–1289.

[9]. Ni, Y.; McVicar, M.; Santos-Rodriguez, R.; Bie, T.D. An End-to-End Machine Learning System for Harmonic Analysis of Music. IEEE Trans. Audio Speech Lang. Process. 2012, 20, 1771–1783.

[10]. Pauwels, J.; Martens, J.P. Combining Musicological Knowledge About Chords and Keys in a Simultaneous Chord and Local Key Estimation System. J. New Music Res. 2014, 43, 318–330.

[11]. Krumhansl, C.L. Cognitive Foundations of Musical Pitch; Oxford University Press: Oxford, UK, 2001; Volume 17.

[12]. Harte, C. Towards Automatic Extraction of Harmony Information from Music Signals. Ph.D. Thesis, Queen Mary University of London, London, UK, 2010.

[13]. Fujishima, T. Realtime Chord Recognition of Musical Sound: A System using Common Lisp Music. In Proceedings of the International Computer Music Conference, Beijing, China, 22–28 October 1999.

[14]. Juslin, P.N.; Sloboda, J. Handbook of Music and Emotion: Theory, Research, Applications; Oxford University Press: Oxford, UK, 2011.

[15]. Dowling, W.J.; Harwood, D.L. Music Cognition; Academic Press: Cambridge, MA, USA, 1986.

[16]. Hatten, R.S. Musical Meaning in Beethoven: Markedness, Correlation, and Interpretation; Indiana University Press: Bloomington, IN, USA, 2004.

[17]. Gómez, E. Tonal Description of Music Audio Signals. Ph.D. Thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2006.

[18]. Tzanetakis, G.; Cook, P.R. Musical Genre Classification of Audio Signals. IEEE Trans. Speech Audio Process. 2002, 10, 293–302.