Chippies NFT: Leveraging Transfer Learning for Solana NFT Sentimental Analysis and Further Market Prediction

Anonymous ACL submission

Abstract

We present Chippies NFT, a first NFT project and research work that incorporates Natural Lanugage Processing into the Solana NFT world. Using self-superivsed training, we introduce a first domain-specific NFT Tweet Language Model that learns a better representation of NFT text and Twitter sentiment. Leveraging transfer learning, the Language Model is trained on a large corpus of pure English text before training on large Twitter's text corpus to understand representation of both English and NFT's social media languages. We also include a novel Hype Metrics function to calculate the current hype of these sentimental embeddings. We will clearly describe our metrics and release the source code of the metric for further studies and benchmarking. On later stages of the project, we are planing to train and release more Deep Learning models that aid NFT traders in making financial decision for any NFT projects.

1 Introduction

Pre-trained language models (LMs) have played an important and novel role in the development of many Natural Language Processing (NLP) systems in recent years. Large pretrained models such as BERT citebert, XL-NET (Yang et al., 2019), ALBERT (Lan et al., 2019), RoBERTa (Liu et al., 2019), GPT-3 (Brown et al., 2020), BART (Lewis et al., 2019), and T5 (Raffel et al., 2019) have become an effective trend in natural language processing. All of these large models adhere to the Transformer architecture proposed by (Vaswani et al., 2017), complete with an attention mechanism. With their large pre-trained checkpoints, the architecture has proven to be very suitable for finetuning downstream tasks using transfer learning. Prior to the advent of large Transformer LMs, traditional wording embedding assigned a fixed global representation to each word. Large pre-trained models can derive word vector representations from large corpora that have been trained.

This will provide the pretrained model with a better understanding of the generalized representation of a trained language/domain, significantly improving performance on downstream finetune tasks. The success of pre-trained models on a generative domain (BERT, RoBERTa, BART, T5, and so on) has paved the way for the development of more specific-domain language models such as codeBERT (Feng et al., 2020) for coding languages, TaBERT citetabert for tabular data, BioBERT (Lee et al., 2019) and PubmedBERT (Gu et al., 2020) for biomedical languages.

Therefore, in the Chippies NFT projects, we introduce a NFT Tweet Encoder BERT, a pre-trained encoder transformer models for NFT Tweet domains. Our model follows the transformer architecture with attention mechanism proposed by Vaswani et al. (2017). The model is trained on a large corpus of tweets and NFT related tweets to understand the language representation of social media tweets and NFT languages. We will then finetune those models on more task-specific domain like sentimental analysis and mood prediction of any NFT Collection's Twitter. As our research works and attempts are novel, we will release a new Hype Metric function to calculate the Hype Score of an output embeddings. This function will be release for both the NFT communities and academia for further studies and improvements.

Having said that, our project is just a first step to incorporate NLP / DL into the Solana NFT world. We will release more studies and NFT-related Deep Learning models for our community as well as AIs incorporation into tokenomics.

In this whitepaper, we offer the following contributions:

- A first pre-trained Encoder Transformer model for NFT Tweet analysing

- A novel Hype Metric function to calculate the Hype Score from Deep Learning's model output embeddings.
- **A pipeline that analyzing public sentiment and mood of any popular or upcoming NFT Collection's Twitter**
- As we are planning to submit our research works to various NLP conference, we will release the training dataset to our communities after got accepted.

2 Chippies Language Model

We explain our models and training procedures we applied in this section

2.1 Model Architecture

We adapted the transformer-based model proposed by Vaswani et al. (2017), the BERT checkpoints and framework [1] implemented by Raffel et al. (2019). Transformer is a breakthrough technology in the field of NLP recently. Transformer alleviates the problem of the all sequential-based model such as LSTM, which allows the model to learn the whole sentences without losing any contextual meaning. Furthermore, it applies the attention mechanism to concentrate on the words where the most relevant information instead of an entire sentence. The original Transformer model contains 6 layers of encoders and decoders, but in our model, we will eliminate the decoders and stack 12 layers of encoders to improve the capability of extracting deep semantic contexts from the sentence. The total number of model parameters is 125M. Below is the architecture of an encoder layer inside our model.

Looking at Figure 1, the input embedding layer will convert a word into a 512-dimension vector representation. After that, the positional encoding layer will apply a sinusoidal function to incorporate the position of the word in the sentence, which is crucial for the model to learn the contextual meaning. The multi-head attention layer will calculate the attention score for each word in the sentence, so that the model will know which words it should concentrate on while learning. The Add & Norm layer will normalize the value of the embeddings by their mean and standard deviation. At the end, these embeddings will be fed through a network

[1] https://github.com/google-research/text-to-text-transfer-transformer

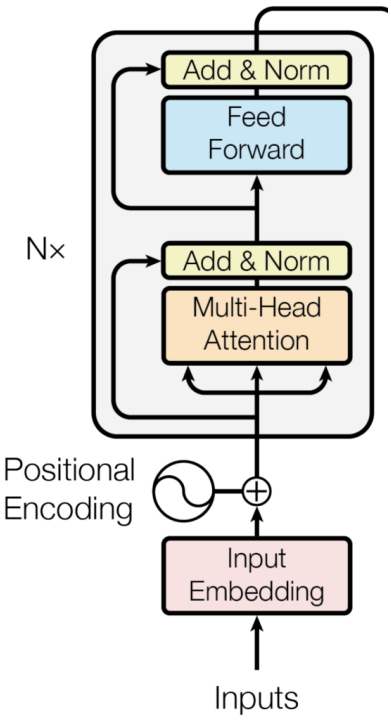

Figure 1: Encoder Layer Architecture

with activation function to have the model learn the essential information.

2.2 Model Fine-Tuning

We will fine-tuning this model on the SemEval 2017 tweets data, the dataset released from International Workshop on Semantic Evaluation. For fine-tuning, we use a normal learning rate of $1e^{-5}$. We will use a batch size of 4 and train the model with 3000 steps of warm-up where it uses a lower learning rate at the beginning. The warm-up steps are reported to improve the performance of the network with the attention mechanism. We trained this model with 50 epochs total.

3 Hype Metric

Before calculate the hype of an NFT collection, we need to define the criteria that account for the hype. We will use 3 main components: the number of followers, the number of tweets and replies, and the sentiment score of all the tweets and replies collecting from an NFT collection's twitter site. The first two components are already provided from the Twitter API. We will use the language model to calculate the third component.

3.1 Collection Sentiment Score

For each tweets and replies, which we will call sentence, collected on the NFT collection's twitter, we will feed them through our Chippies Language Model. For each of the sentence it will output 3 sentimental metrics: Positive, Neutral, and Negative. Those 3 metrics will add up to 1.

$$ChippiesModel(sentence) = \begin{bmatrix} pos \\ neu \\ neg \end{bmatrix}$$

$$pos + neu + neg = 1$$

After gathering all the sentiment metrics for each tweets, let n be the total number of tweets and replies, we will calculate the overall sentiment score of a collection following the equation in the Figure 2. Then we will normalize the calculated score to $0 - 1$ range.

$$\text{Score} = \frac{1}{n} \sum_{i=1}^{n} \begin{bmatrix} pos_i \\ neu_i \\ neg_i \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

Figure 2: NFT Collection Sentiment Score Function

3.2 Overall Collection Score

We will use the collection sentiment score calculated in the previous section to calculate the overall collection score by substituting it in the formula in Figure 3.

$$Score = \begin{bmatrix} 1000 \times sentiment \\ \#followers \\ \#tweets \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

Figure 3: NFT Collection Overall Score Function

3.3 Normalized Hype Score

After calculating the overall collection score, we want to represent those scores in range of $0 - 100$ to assist users' with interpretation. For this purpose, we will use logistic function to normalize our overall collection score, since logistic function's value is always in the $[0, 1]$ bound. Then we will multiply those normalized score with 100 to rescale it to the $0 - 100$ range. Moreover, we don't want the hype score grows linearly with the overall collection score. Logistic function has non-linear slope. The relationship between the input and output is inversely proportional, as the input rate of change increases, the output rate of change decreases. It fits with the characteristic of the NFT market, where there might be some difference between hundreds followers collections compare to thousands followers ones in the extent of hype, but when comparing hundreds thousands collections versus hundreds thousands ones, they are all hype collections that we should consider to look at, so the number of followers becomes less matter. The slope of the logistic function satisfies the former characteristic perfectly.

$$f(x) = \frac{1}{1 + e^{-k(x - x_0)}}$$

where

x : the overall collection score
k : the steepness of the slope
x_0 : the value of the logistic graph's midpoint

Figure 4: Normalized Logistic Function

4 Road Map

Having said that, we will illustrate our road map with full features of AI tools, and organization dashboard to aid Solan NFT trader in making any financial decision.

4.1 Phase 1

An equipment package changing the entire the user experience:

Sentimental analysis:

- Sentimental analysis of twitter upcoming/new/popular nft collections.
- Deep Learning models and pipeline analyze twitter interactions and activity.
- Provides hype score.

Wallet organization:

- Take all statistics on Magic Eden/Solscan
- Connects wallet to spreadsheet and portrays each transaction in organized matter

4.2 Phase 2

Generative Adversarial Networks (GAN):

- GANs allow for ANYBODY to be a creator.
- If you have an idea and you can't draw, write out your idea and do a rough sketch. GANs will take what you gave it and generate a unique image. This can also be used for artists to enhance their images. With this art we might even allow some holders to release with us and earn royalties.
- Staking with our own unique tokenomics and NFTs.
- Our token will be used to evolve your chippie.

5 Acknowledgement

This project is fully initiated, developed, and maintained by our passionate engineering team including engineers at: Coinbase, Google, Snapchat, LinkedIn, Vistaprint, and Snowflakes.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. Codebert: A pre-trained model for programming and natural languages. *CoRR*, abs/2002.08155.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *CoRR*, abs/2007.15779.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *CoRR*, abs/1901.08746.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.