International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Applying Predictive Prefetching Using Deep Learning and Nlp to Shorten Web Page Loading Times

Dr. Aarti Sehwag[1], Akshay Prabhat Mishra[2], Dhiraj Kargeti[3], Dhruv Pahadia[4], Gaurav Mangal[5]

[1]Assistant Professor, Computer Science and Engineering, Bharati Vidyapeeth College of Engineering
[2,3,4,5]Student, Computer Science and Engineering, Bharati Vidyapeeth College of Engineering

Abstract
In response to the growing demand for seamless user experiences in web applications, predictive prefetching based on user interaction has become a crucial technique to alleviate perceived latency. This paper presents an overview of predictive prefetching based on user interaction for web applications, elucidating essential concepts such as user interaction analysis, prediction algorithms, resource preloading, adaptability, and learning. The primary objective is to reduce perceived latency and improve the responsiveness of web applications by strategically preloading resources, thereby optimizing the user experience.

Keywords: N-Gram, Tokenization, Prefetching, Sequential, Predict, User Navigation Patterns

1. Introduction

Predictive prefetching, a sophisticated technique rooted in the realms of artificial intelligence and machine learning, has emerged as a transformative paradigm in the domain of computer systems and data management. As the digital landscape continues to burgeon with vast amounts of data, the demand for efficient data retrieval mechanisms becomes paramount.

Predictive prefetching addresses this need by leveraging the capabilities of deep learning algorithms to anticipate future data access patterns. The first model employs a sequence-to-sequence prediction approach, leveraging Long Short-Term Memory (LSTM) networks. This model treats a user's journey as a sequential pattern, capturing intricate dependencies within the data and enabling the anticipation of subsequent actions. The sequence-to-sequence architecture provides a foundational understanding of how users navigate through a website, facilitating the prediction of the next sequence of interactions. In parallel, the second model adopts an NLP-based LSTM for string prediction.

This model focuses on predicting the next string in the user's interaction pattern, utilizing natural language processing techniques to discern underlying linguistic patterns. By treating user interactions as strings and leveraging the power of LSTM networks, this model augments our understanding of user behavior, providing insights into the linguistic nuances that influence their navigation choices.

2. Model 1: Sequence to Sequence

The Seq2Seq model with LSTM architecture is designed to understand and predict the sequential patterns of user interactions on a website. The model comprises an embedding layer to convert categorical data into numerical representations, an LSTM layer to capture temporal dependencies, and a dense layer for sequence prediction. The input sequence is fed into the embedding layer, followed by the LSTM layer, which encodes the input sequence into a fixed-size context vector. The decoder LSTM then generates the output sequence based on this context vector.

Figure 1: Seq2Seq Model Architecture

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 4, 50)             300

 lstm (LSTM)                 (None, 4, 100)            60400

 dense (Dense)               (None, 4, 6)              606

=================================================================
Total params: 61,306
Trainable params: 61,306
Non-trainable params: 0
_________________________________________________________________
```

Despite its sophisticated architecture, the Seq2Seq model faces challenges contributing to its lower accuracy. One primary challenge arises from the dataset, where issues such as data quality, limited diversity in user behaviors, and class imbalance impact the model's ability to generalize effectively. The complex nature of user interactions on websites, coupled with the potential noise in the data, makes it challenging for the model to discern meaningful patterns. Additionally, the imbalance in the distribution of user interaction sequences may lead to biased learning, affecting the model's predictive capabilities.

The dataset used for training and validation consists of sequences representing user interactions on a website. Each sequence is a chronological sequence of page visits or interactions, denoted by unique URLs. The dataset is preprocessed to convert URLs into numerical indices for model compatibility. While the dataset provides valuable insights into user behavior, its limitations, such as data quality issues and potential biases, impact the model's overall performance.

Figure 2: User Dataset

```
Algorithm -> Branch -> Dependency -> UnitTesting -> Heap
VersionControl -> Library -> Function -> VersionControl -> Varia
Profiling -> GUI -> Event -> Git -> ContinuousIntegration
GarbageCollection -> API -> Conditional -> Library -> Stack
Function -> Dependency -> Algorithm -> Merge -> Repository
```

3. Model 2: NLP/N-Gram based string prediction using LSTM

Figure 3: N-Gram Model Architecture

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 2645, 100)         7300

 lstm_1 (LSTM)               (None, 150)               150600

 dense_1 (Dense)             (None, 73)                11023

=================================================================
```

The NLP-based LSTM model is crafted to predict the next string in the user's interaction pattern, treating user interactions as linguistic sequences. The architecture involves an embedding layer for converting strings into numerical representations, an LSTM layer to capture linguistic patterns, and a dense layer for string prediction. The model aims to discern subtle linguistic nuances within user interaction sequences, contributing to a more comprehensive understanding of user behavior.

The text data is tokenized using the Keras Tokenizer class, and the total number of unique words is determined. N-Gram sequences are then constructed from the tokenized text, where each sequence represents an evolving linguistic context by incrementally adding tokens.

TOKENIZATION:

```
{
        'ajax': 1,
        'memory': 2,
        'security': 3,
        'nosql': 4,
        'merge': 5,
        'performance': 6,
        'markup': 7
}
```

An N-Gram model is a probabilistic language model that's built by counting how often word sequences occur in a text corpus. The model then estimates the probabilities of these sequences.

4. Result and Analysis

4.1 Sequence-to-Sequence Model:

The Sequence-to-Sequence model, designed to predict user navigation patterns through a website, presents intriguing insights into the challenges associated with this particular predictive task.

4.1.1 Model Performance:

The model's accuracy, while indicative of its ability to learn certain patterns, reveals limitations in capturing the nuanced dependencies within user navigation sequences.

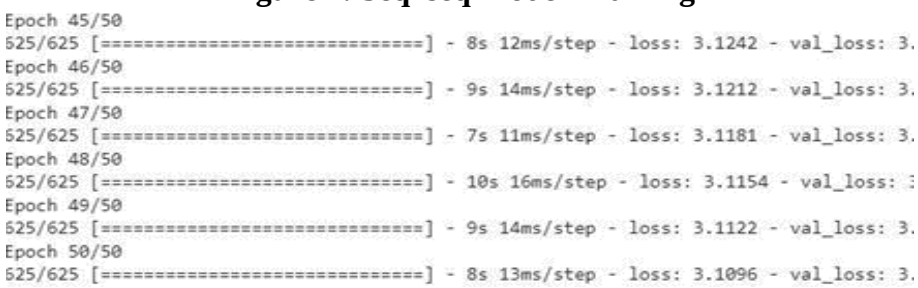
The accuracy metrics, evaluated on a validation set, demonstrate a notable gap from optimal performance.

4.1.2 Training

The model ran for 50 epochs with a batch size of 64. Although the overall accuracy is far from optimal, this stands to show the inability of LSTM based models to predict the sequences in string format.

The website interaction dataset exhibits a wide range of user behaviors, making it challenging for the model to generalize effectively. Certainly navigation paths are more prevalent than others, leading to imbalanced class distributions. This imbalance poses challenges for the model in learning infrequent patterns.

Figure 4: Seq2seq Model Training



4.1.3 Predictions

Figure 7: Seq2Seq Model Predictions

```
1/1 [==============================] - 0s 54ms/step - loss: 5
1/1 [==============================] - 1s 882ms/step
['API', 'Heap', 'Profiling', 'ContinuousIntegration']
```

4.2 NLP-Based LSTM Model:

Contrasting with the Sequence-to-Sequence model, the NLP-based LSTM model, designed for predicting the next string in user interaction sequences, showcases notable improvements in accuracy and pattern learning. The NLP-based LSTM model excels in capturing linguistic nuances within user interaction sequences, resulting in significantly improved accuracy metrics compared to the alternative model.

4.2.1 Model Performance:

The NLP-based LSTM model excels in capturing linguistic nuances within user interaction sequences, resulting in significantly improved accuracy metrics compared to the alternative model. There is a significant improvement in the accuracy as well as the robustness in the model.

4.2.2 Training

The model ran for 25 epochs but with a much better accuracy. The overall accuracy has shown a great improvement with a significant decrease in loss metrics.

Figure 6: N-Gram Model Training

```
43/43 [==============================] - 8s 185ms/step - loss: 0.2396 - accuracy: 0.9825
Epoch 15/25
43/43 [==============================] - 6s 146ms/step - loss: 0.2209 - accuracy: 0.9818
Epoch 16/25
43/43 [==============================] - 9s 212ms/step - loss: 0.2041 - accuracy: 0.9840
Epoch 17/25
43/43 [==============================] - 6s 146ms/step - loss: 0.1901 - accuracy: 0.9840
Epoch 18/25
43/43 [==============================] - 9s 204ms/step - loss: 0.1758 - accuracy: 0.9840
Epoch 19/25
43/43 [==============================] - 8s 181ms/step - loss: 0.1653 - accuracy: 0.9825
Epoch 20/25
43/43 [==============================] - 7s 163ms/step - loss: 0.1555 - accuracy: 0.9840
Epoch 21/25
43/43 [==============================] - 8s 195ms/step - loss: 0.1438 - accuracy: 0.9854
Epoch 22/25
43/43 [==============================] - 7s 149ms/step - loss: 0.1361 - accuracy: 0.9832
Epoch 23/25
43/43 [==============================] - 8s 184ms/step - loss: 0.1280 - accuracy: 0.9854
Epoch 24/25
43/43 [==============================] - 6s 141ms/step - loss: 0.1209 - accuracy: 0.9840
Epoch 25/25
43/43 [==============================] - 8s 180ms/step - loss: 0.1148 - accuracy: 0.9854
```

4.2.3 Predictions

Figure 7: N-Gram Model Predictions

```
[21, 14, 10, 23]
1/1 [==============================] - 0s 444ms/step
[21, 14, 10, 23, 43]
1/1 [==============================] - 0s 460ms/step
[21, 14, 10, 23, 43, 51]
1/1 [==============================] - 1s 621ms/step
[21, 14, 10, 23, 43, 51, 69]
1/1 [==============================] - 0s 459ms/step
[21, 14, 10, 23, 43, 51, 69, 20]
1/1 [==============================] - 0s 452ms/step
[21, 14, 10, 23, 43, 51, 69, 20, 7]
1/1 [==============================] - 0s 451ms/step
Dependency Function CSS Scripting unittesting rest devops agile markup apiendpoint
```

5. Conclusion

In conclusion, this research underscores the importance of task-specific model design and opens avenues for future investigations. While each model provides distinct insights, the amalgamation of diverse modeling techniques holds promise for advancing user interaction analysis. This exploration not only contributes to the current understanding of deep learning applications but also sets the stage for continued advancements in predictive modeling for user-centric applications.

6. References

1. Koch, C., Lins, B., Rizk, A., Steinmetz, R., & Hausheer, D. (2017). "vFetch: Video prefetching using pseudo subscriptions and user channel affinity in YouTube." In 2017 13th International Conference on Network and Service Management (CNSM) (pp. 1-8). IEEE. DOI: 10.23919/CNSM.2017.8256011
2. Liu, X., Ma, Y., Liu, Y., & Wang, X. (October 2017). "SWAROVski: Optimizing Resource Loading for Mobile Web Browsing." IEEE Transactions on Mobile Computing, PP(99), 1-1. DOI: 10.1109/TMC.2016.2645563
3. Liu, J., Hu, Z., Guo, J., & Hao, H. (2017). "An Internet Application-Driven Cache Placement Algorithm for Software-Defined Information-Centric Networking." In 2017 International Conference on Computer Technology, Electronics and Communication (ICCTEC) (pp. 1-5). IEEE. DOI: 10.1109/ICCTEC.2017.00301
4. Jonassen, S., Cambazoglu, B. B., & Silvestri, F. (August 2012). "Prefetching query results and its impact on search engines." DOI: 10.1145/2348283.2348368
5. Joo, M., An, Y., Roh, H., & Lee, W. (March 2021). "Predictive Prefetching Based on User Interaction for Applications." IEEE Communications Letters, 25(3), 821-824. DOI:10.1109/LCOMM.2020.3038255
6. Zhou, Y., Philbin, J., & Li, K. (June 2001). "The Multi-Queue Replacement Algorithm for Second Level Buffer Caches." In Proceedings of the General Track: 2001 USENIX Annual Technical Conference (pp. 91–104). DOI: https://doi.org/10.1145/3572778
7. Vaithinathan, K., Pernabas, J. B., Parthiban, L., Shrestha, B., Joshi, G. P., & Moon, H. (August 2021). "An Improved Web Caching System With Locally Normalized User Intervals." IEEE Access, 9, 112490-112501. DOI: 10.1109/ACCESS.2021.3103804
8. Leith, D. J. (March 2021). "Web Browser Privacy: What Do Browsers Say When They Phone Home?" IEEE Access, 9, 41615-41627. DOI: 10.1109/ACCESS.2021.3065243
9. Wang, H., Li, Y., Jin, D., & Han, Z. (July 2021). "Attentional Markov Model for Human Mobility0 Prediction." IEEE Journal on Selected Areas in Communications, 39(7), 2213-2225. DOI: 10.1109/JSAC.2021.3078499
10. Zhang, M., Tang, Q., Kim, J.-G., Burgstaller, B., & Kim, S.-D. (2023). "Adaptive Regression Prefetching Algorithm by Using Big Data Application Characteristics." Applied Sciences, 13(7), 4436. DOI: 10.3390/app13074436
11. Fang, J., Xu, Y., Kong, H., & Cai, M. (July 2023). "A Prefetch Control Strategy Based on Improved Hill-Climbing Method in Asymmetric Multi-Core Architecture." The Journal of Supercomputing, 79(10), 10570–10588. DOI: 10.1007/s11227-023-05078-6