Expanding Embedding Spaces

Charles Curt
charles@sliced-ai.com

June 2024

Abstract

Long-running tasks typically involve processing highly similar information over time, which poses a significant challenge for Large Language Models (LLMs) in terms of storage and retrieval. Retrieval-augmented generation (RAG) systems, while effective in some scenarios, rely on static embedding spaces that do not expand with new data, leading to poor data retrieval performance in long-term processes.
Continual learning and memory growth are essential for enabling LLMs to handle these evolving long-running tasks. By expanding the embedding space, LLM agents can better differentiate between similar data points, allowing for more accurate classification and retrieval. This work explores the mechanisms behind embedding space expansion using autoencoders and progressive training, with the Critical Role Dataset (CRD3) as a testbed. Learning how to expand embedding spaces is crucial for any long-running LLM application, whether through RAG or continual learning systems.

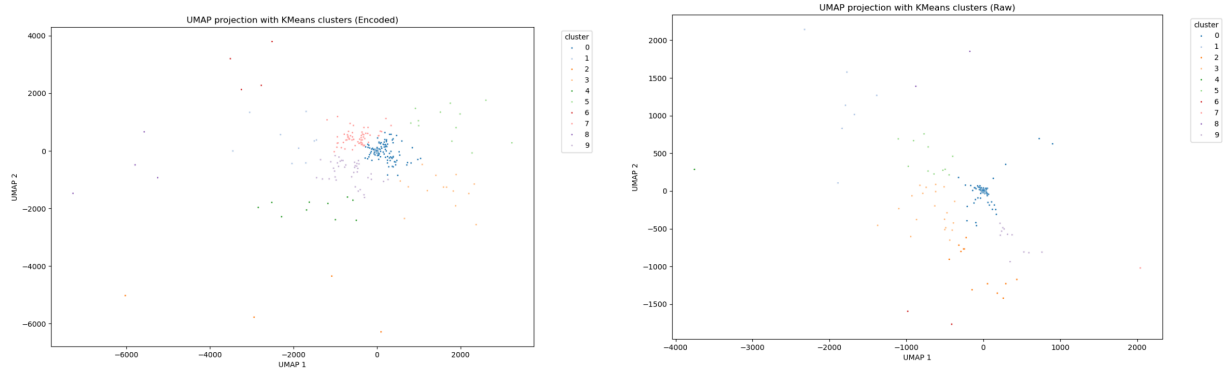
1 Introduction

Large Language Models (LLMs), such as GPT-3 and BERT, have demonstrated significant success across various natural language processing (NLP) tasks. As these models evolve into autonomous agents, particularly for long-running and continuous tasks, one of the major challenges is embedding space management. In such tasks, the model's ability to continue learning and expanding its understanding of new information is paramount.

1.1 Embedding Spaces in Long-Running Tasks and Continual Learning

When LLM agents engage in long-running tasks, they often process highly similar information over time. As new data points are added, if all of this information is placed into the same limited embedding space, the pre-trained model may struggle to differentiate between the new and existing data. This is especially problematic when the differences between data points are subtle or when new data closely resembles previously processed information. Pre-trained models tend to collapse similar data into overlapping clusters, making it difficult to extract meaningful insights from new inputs.

This issue directly relates to how LLM agents store information in a continual learning scenario. In a non-continual learning model such as RAG, the embedding space is static and does not evolve or expand as new information is introduced. This limits the systems ability to handle highly similar data over time because the embedding space becomes crowded, resulting in errors during retrieval or decision-making processes. In contrast, a continual learning agent



(a) Autoencoder-Expanded Embeddings

(b) Raw OpenAI Embeddings

Figure 1: Core Finding: Comparison of UMAP Projections. Raw OpenAI Embeddings vs Autoencoder-Expanded Embeddings. This shows the significant improvement in embedding space structure after using an autoencoder.

would need an expanding embedding space that mirrors how it stores and recalls information internally, adapting dynamically as it encounters new data.

Expanding the embedding space becomes crucial not only for distinguishing subtle differences in new information but also for supporting continual learning in LLM agents. By allowing more granularity and separation between data points, an LLM agent can better discern the subtle distinctions that are critical in long-running tasks. This expanded space is necessary for an agent to understand the variety of actions it can take, even when confronted with near-identical inputs in the same task environment. Without expanding the space, LLM agents or RAG systems would struggle with generalization and retrieval as their knowledge base grows.

Therefore, the primary objective of this work is to explore methods for expanding embedding spaces, which is directly correlated with the capabilities needed for continual learning in LLM agents. By focusing on highly similar text data, this research sheds light on how embedding spaces can be managed to support long-running processes without overwhelming the system with overlapping information.

2 Dataset

2.1 Critical Role Dataset (CRD3)

The Critical Role dataset (CRD3) was chosen as the primary testbed for this study. This dataset is collected from 159 episodes of the live-streamed show Critical Role, where a group of players collaboratively engage in a Dungeons and Dragons campaign. The dataset consists of 398,682 turns of dialogue transcribed into text and includes corresponding abstractive summaries collected from the Fandom wiki.

Critical Role is an ideal dataset for this study due to its unique characteristics:

- It is a long-running task involving a continuous narrative, making it an excellent candidate for studying how embedding spaces behave with long-term, highly similar data.
- The narratives and dialogues are generated through spoken interaction, with frequent repetition and similarity in phrasing, making it well-suited for exploring the challenges of expanding embedding spaces to avoid data overlap.

The linguistic uniqueness of this dataset lies in the

collaboration between players, with multiple abstractive summaries and strong semantic ties between dialogues. This allows the study of how an evolving embedding space can better differentiate between very similar data points in a long-running task like Critical Role.

3 Methodology

3.1 Embedding Models and Architectures

The following key models and approaches were utilized in this study:

- **BERT Embeddings**: Used as a baseline to observe how traditional pre-trained embeddings handle highly similar text data.
- **Autoencoders**: These were applied to expand the embedding space by learning compressed representations of the data and then reconstructing them, thereby dispersing overlapping data points.
- **Progressive Training**: Progressive training was employed to incrementally expand the embedding space by gradually increasing the dataset size and the number of training epochs. The data was shuffled as the size increased, ensuring better generalization and preventing overfitting.

3.2 Training Setup

The training setup consisted of two primary configurations:

- **Progressive Training Configuration**: The initial dataset size was set to 1024, with an increment ratio of 10% and up to 50 epochs. Learning rates and batch sizes were dynamically adjusted based on the models performance. Data was shuffled as the size increased to prevent overfitting.
- **Autoencoder Configuration**: The initial dataset size was set to 12,288, with an increment ratio of 5% and a maximum of 5,000 epochs. The autoencoder's goal was to maximize the dispersion of data points in the embedding space while maintaining the integrity of the learned representations.

Key raw data collected during training:

- **Progressive Training (50 epochs)**: Train Loss: 0.00026219, Val Loss: 0.00025741, Train Similarity: 77.03%, Val Similarity: 77.52%
- **BERT Embedding Training Results (33 epochs)**: Train Loss: 0.00054292, Val Loss: 0.00052701, Train Similarity: 41.12%, Val Similarity: 43.65%
- **Autoencoder Progressive Training Results**: Data Size: 14,224, Train Loss: 0.00217701, Val Loss: 0.00142907, Train Similarity: 2.01%, Val Similarity: 3.93%

4 Experiments and Results

4.1 UMAP Projection of BERT Embeddings

The first figure presents the UMAP projection of BERT embeddings from the pre-trained BERT model applied to the Critical Role dataset (CRD3). The figure reveals that the embeddings were clumped together, highlighting the inability of BERT to distinguish between highly similar data points.

4.2 PCA Projections of Raw vs Autoencoder-Expanded Embeddings

This set of figures shows PCA projections of the raw OpenAI embeddings and the embeddings after the autoencoder expansion. PCA provides another method for visualizing the embedding space. The

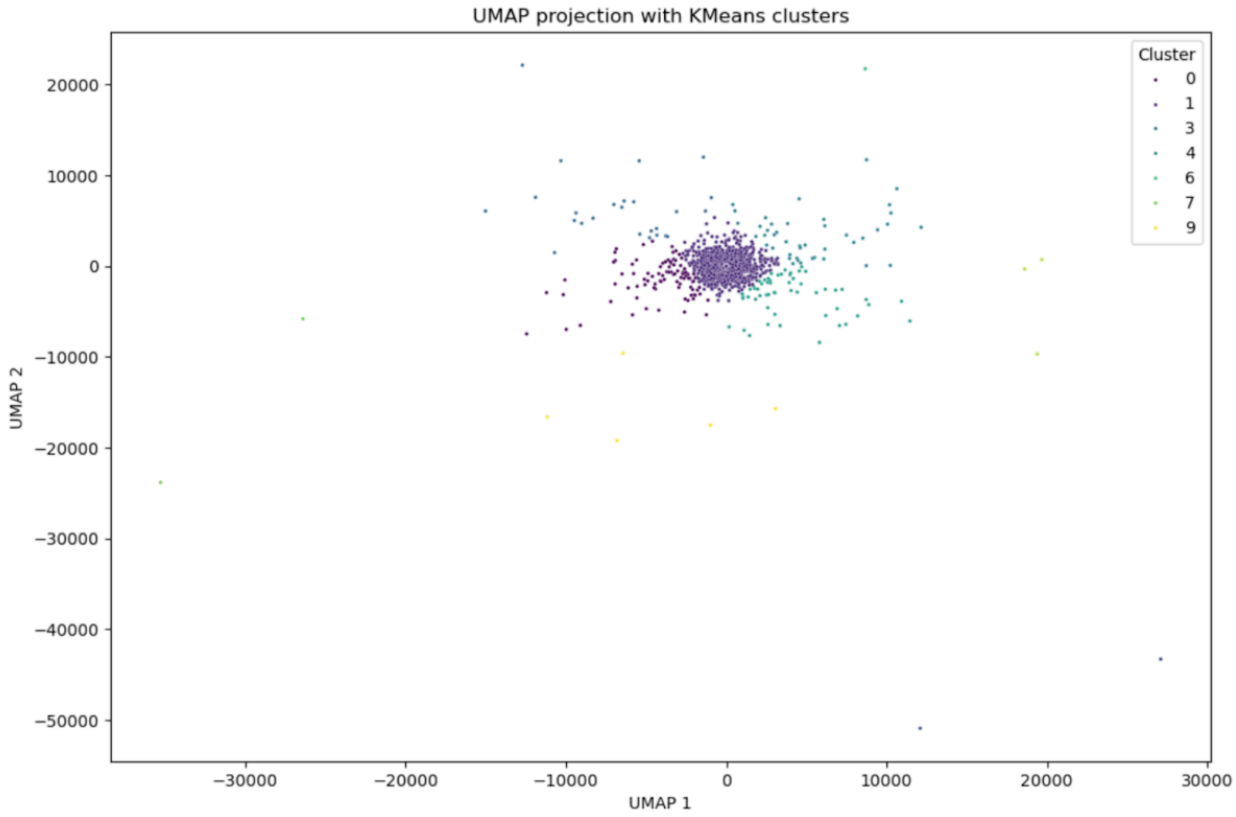


Figure 2: UMAP Projection of Initial BERT Embeddings Showing Clustering.

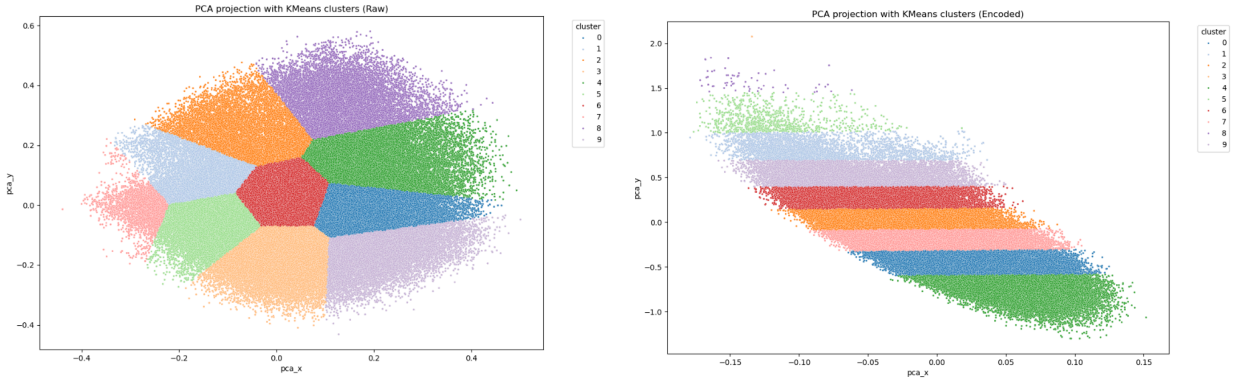
raw OpenAI embeddings show no structure, while the post-autoencoder embeddings reveal clear structure.

4.3 Comparison of Progressive vs Standard Training Loss Curves

This figure shows the training loss curves for progressive training (with and without shuffling) compared to standard training. It highlights how progressive training prevents overfitting and leads to better long-term convergence, even though it takes longer. Progressive training was performed to prevent overfitting, as in real-world scenarios like continual learning or RAG, it's often impractical to leave out data for validation, so this method provides confidence that overfitting is minimized.

4.4 UMAP Visualization Before and After High-Accuracy Autoencoder Training

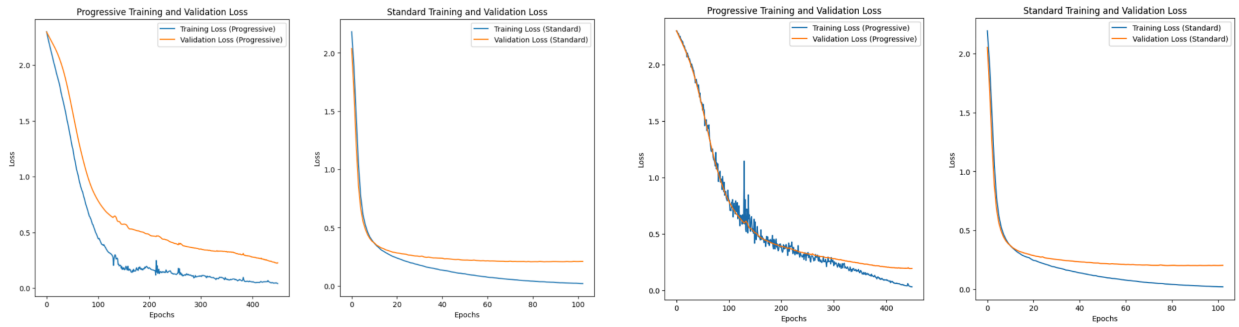
This figure shows the UMAP projections of the embedding space before and after training a high-accuracy autoencoder (trained to ¿95% similarity). It demonstrates how high-accuracy training expands the embedding space while maintaining separation between data points.



(a) Raw OpenAI Embeddings (PCA)

(b) Autoencoder-Expanded Embeddings (PCA)

Figure 3: PCA Projections: Raw OpenAI Embeddings vs Autoencoder-Expanded Embeddings.



(a) Without Random Shuffling

(b) With Random Shuffling

Figure 4: Training Loss Curves for Progressive Training vs Standard Training.

4.5 Progressive Training Similarity vs Loss (Intermediate Stage)

This figure shows the similarity and loss values during progressive training at an intermediate stage (around 50 epochs). It demonstrates the steady improvement of the training process and the balanced expansion of the embedding space without significant loss degradation.

5 Discussion

The experiments produced several key findings that highlight the importance of expanding embedding spaces in long-running processes:

- **Progressive Training** was found to be effective at expanding the embedding space over time, although it came at the cost of slower convergence compared to standard methods. Random shuffling further improved the performance of progressive training.
- **Autoencoders** successfully dispersed similar

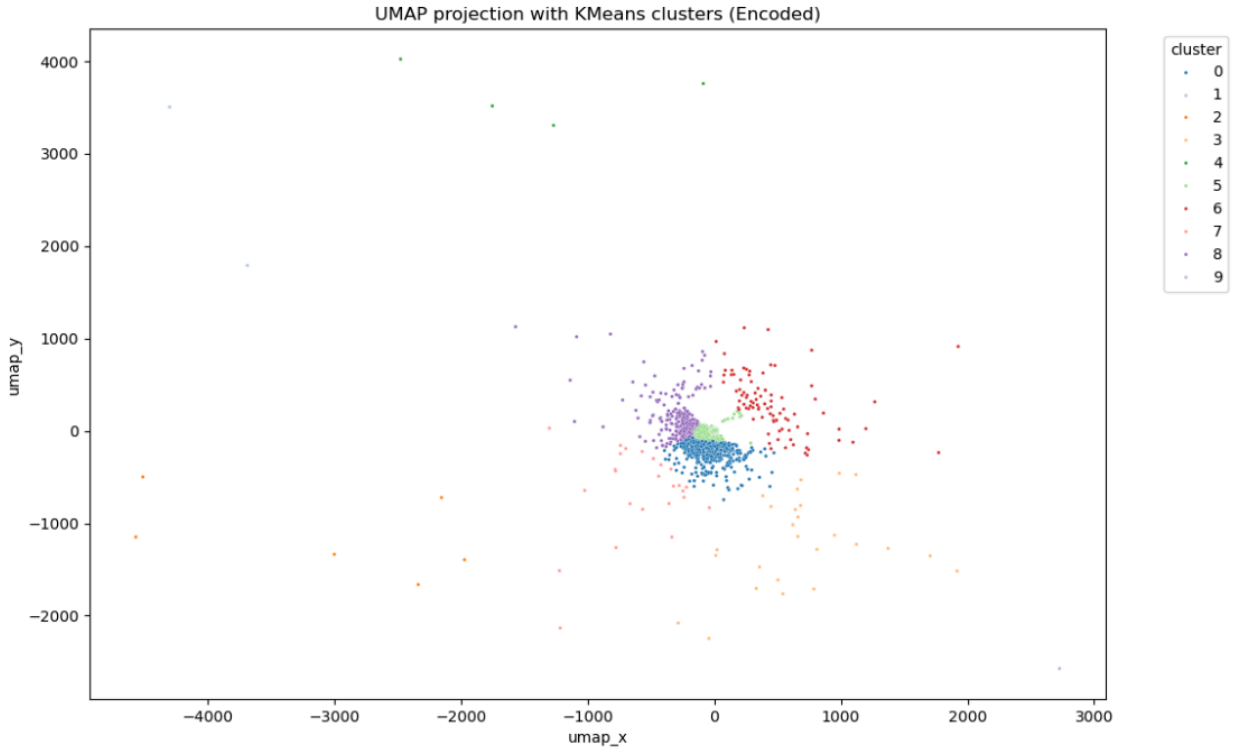


Figure 5: UMAP Visualization of Embedding Space Before and After High-Accuracy Autoencoder Training.

data points in the embedding space, though fine-tuning was necessary to avoid introducing noise into the expanded space.

5.1 Noise and Collapse in Embedding Spaces: The Role of Progressive Training

In the context of embedding space expansion, noise and collapse are two critical concerns. Overfitting can lead to noise, where the embedding space becomes overly complex and starts capturing irrelevant features, while underfitting leads to collapse, where the embedding space becomes too diffuse and fails to distinguish between important data points.

Progressive training was chosen as a method to balance between overfitting and underfitting. By incrementally increasing the training dataset size and shuffling the data at each stage, progressive training ensures that the model does not become overly specialized on any subset of data, thus preventing overfitting and the resulting noise. At the same time, it prevents collapse by ensuring that the model is trained on an expanding dataset, allowing the embedding space to grow and adapt to new information without losing important distinctions.

This approach is particularly important in real-world scenarios like continual learning and RAG, where it is often impractical to leave out data for validation. Progressive training provides a mechanism that allows the model to grow its knowledge base without overfitting, maintaining the quality of the embedding space and ensuring that it continues to serve its in-

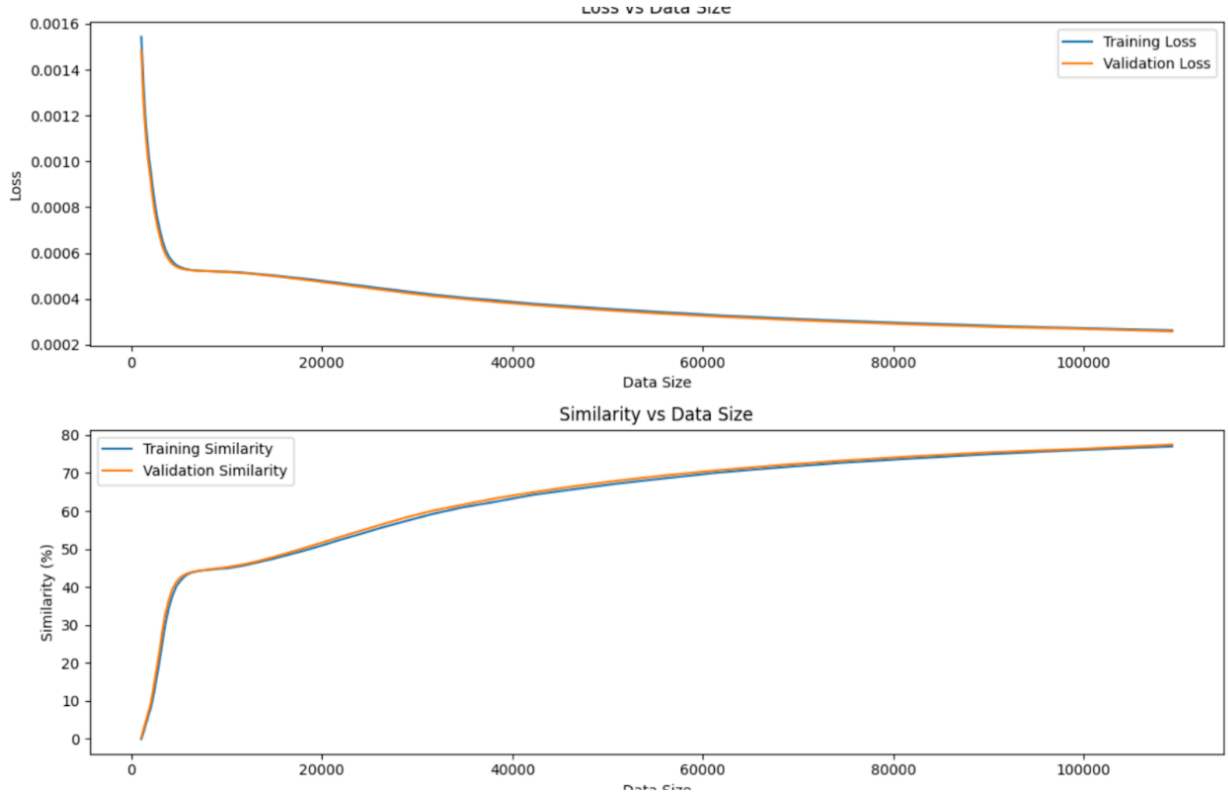


Figure 6: Similarity and Loss Curves at Epoch 50 in Progressive Autoencoder Training.

tended purpose over time.

6 Conclusion

This research demonstrates that methods such as autoencoders and progressive training can successfully expand the embedding space for highly similar text data, as evidenced by the experiments conducted on the Critical Role Dataset (CRD3). The ability to expand the embedding space is directly correlated with the needs of continual learning in LLM agents, which must store and adapt information over time. Without this expansion, non-continual learning models like RAG fail to maintain the separation and structure required for long-term processing of highly similar data.

Future work will focus on taking the lessons learned from embedding space expansion and applying them to continual learning LLM agents. This will allow the development of more sophisticated models that can manage long-running tasks in dynamic environments without succumbing to overlapping information or retrieval errors.

References

The following references are used across multiple papers related to continual learning and developing large language models (LLMs) for long-running tasks. While not all are directly part of this paper, they provide a comprehensive background for understanding these interconnected topics.

- Chen, Y., et al. (2024). Improving language plasticity via pretraining with active forgetting. *NeurIPS 2023*. arXiv:2307.01163. https://doi.org/10.48550/arXiv.2307.01163
- Shumailov, I., et al. (2023). The curse of recursion: Training on generated data makes models forget. *arXiv:2305.17493*. https://doi.org/10.48550/arXiv.2305.17493
- Kaplan, J., et al. (2020). Scaling laws for neural language models. *arXiv:2001.08361*. https://doi.org/10.48550/arXiv.2001.08361
- Mnih, V., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533. https://doi.org/10.1038/nature14236
- Ibrahim, A., et al. (2024). Simple and scalable strategies to continually pre-train large language models. *arXiv:2403.08763*. https://doi.org/10.48550/arXiv.2403.08763
- Mireshghallah, F., et al. (2022). Memorization in NLP fine-tuning methods. *arXiv:2205.12506*. https://doi.org/10.48550/arXiv.2205.12506
- Alemohammad, S., et al. (2023). Self-consuming generative models go MAD. *arXiv:2307.01850*. https://doi.org/10.48550/arXiv.2307.01850
- Mnih, V., et al. (2013). Playing Atari with deep reinforcement learning. *arXiv:1312.5602*. https://doi.org/10.48550/arXiv.1312.5602
- Gekhman, Z., et al. (2024). Does fine-tuning LLMs on new knowledge encourage hallucinations? *arXiv:2405.05904*. https://doi.org/10.48550/arXiv.2405.05904
- Ha, D., and Schmidhuber, J. (2018). World models. *arXiv:1803.10122*. https://doi.org/10.48550/arXiv.1803.10122
- Das, A., et al. (2024). A decoder-only foundation model for time-series forecasting. *ICML 2024*. Google Research. https://research.google/blog/a-decoder-only-foundation-model-for-time-series-forecasting/
- Parisi, G. I., et al. (2018). Continual lifelong learning with neural networks: A review. *arXiv:1802.07569*. https://doi.org/10.48550/arXiv.1802.07569
- Hernandez, D., et al. (2022). Scaling laws and interpretability of learning from repeated data. *arXiv:2205.10487*. https://doi.org/10.48550/arXiv.2205.10487
- Madaan, D., et al. (2021). Representational continuity for unsupervised continual learning. *arXiv:2110.06976*. https://doi.org/10.48550/arXiv.2110.06976
- Zhou, H., et al. (2022). Fortuitous forgetting in connectionist networks. *arXiv:2202.00155*. https://doi.org/10.48550/arXiv.2202.00155
- Kirkpatrick, J., et al. (2017). Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13), 3521-3526. https://doi.org/10.48550/arXiv.1612.00796