LinkedIn Post Embeddings: Industrial Scale Embedding Generation and Usage across LinkedIn

Sudarshan Srinivasa Ramanujam[*]
LinkedIn Corporation
Mountain View, CA, USA
sramanujam@linkedin.com

Akanksha Bindal[*]
LinkedIn Corporation
Mountain View, CA, USA
abindal@linkedin.com

Yu Jiang[*]
LinkedIn Corporation
Mountain View, CA, USA
tjiang@linkedin.com

Timothy J. Hazen[*]
LinkedIn Corporation
Mountain View, CA, USA
thazen@linkedin.com

David Golland[*]
LinkedIn Corporation
Mountain View, CA, USA
dgolland@linkedin.com

Fengyu Zhang
LinkedIn Corporation
Mountain View, CA, USA
fezhang@linkedin.com

Daqi Sun
LinkedIn Corporation
Mountain View, CA, USA
daqsun@linkedin.com

Wanning Li
LinkedIn Corporation
Mountain View, CA, USA
wannli@linkedin.com

Birjodh Singh Tiwana
LinkedIn Corporation
Mountain View, CA, USA
btiwana@linkedin.com

Siddharth Dangi
LinkedIn Corporation
Mountain View, CA, USA
sdangi@linkedin.com

Peng Yan[†]
LinkedIn Corporation
Mountain View, CA, USA
pyan@linkedin.com

Abstract

A post embedding (representation of text in embedding space that effectively captures semantic meaning) is a foundational component of LinkedIn that is consumed by product surfaces in retrieval and ranking (e.g., ranking posts in the feed or video tab). This paper presents the post embeddings used at LinkedIn, where a pre-trained transformer-based large language model (LLM) is taken as input and fine-tuned using multi-task learning across a diverse set of semantic labeling tasks. We observe positive transfer, leading to improved performance across all tasks, compared to training them independently. The generated post embeddings outperform baseline models in zero-shot learning, demonstrating its potential for broader applicability. Furthermore, the generated post embeddings' performance surpasses that of OpenAI's ADA-001 and ADA-002 embeddings on LinkedIn specific datasets and tasks. We also describe the offline evaluation methodology and the deployment to our nearline infrastructure, which makes the post embedding available for use within minutes of post creation for any downstream application. We present how the embeddings were applied in the Feed product surface, in both ranking and retrieval stages, and showcase the real world online impact to demonstrate the superior performance of these embeddings. Finally, we also share the results of applying the embeddings to the retrieval system of our video ranking product surface in LinkedIn. These embeddings have been battle-tested in production at LinkedIn for over two years, consistently powering multiple products.

[*]All authors contributed equally to this research.
[†]LinkedIn Alumni

Accepted for presentation at the 34th ACM International Conference on Information and Knowledge Management (CIKM 2025).

1 Introduction

LinkedIn is the world's largest professional network, connecting over a billion users across 200+ countries and territories [2, 3]. Our platform fosters a thriving information exchange ecosystem, helping members discover valuable content, learn new skills, and explore career opportunities. Among various content types, posts play a crucial role in facilitating knowledge sharing between creators and consumers. To enhance content discovery and engagement, LinkedIn introduced out-of-network content recommendations, significantly expanding the pool of candidate posts. This shift made efficient embedding-based retrieval (EBR) across a vast corpus essential for delivering high-quality recommendations. Consequently, state-of-the-art recommendation models require rich, semantically meaningful representations of posts to improve content understanding and ranking. Prior work has explored various approaches for converting text into embeddings that effectively capture semantic meaning. These methods range from traditional word vector models like Word2Vec and GloVe to more advanced transformer-based models such as BERT[6], RoBERTa [8], T5 [12] and more recently, to OpenAI's models such as ADA-002 [7]. These newer OpenAI embedding models are designed for out-of-the-box usage, requiring no fine-tuning, and are intended to perform effectively in zero-shot learning scenarios. In this paper, we introduce LinkedIn Post Embedding model which is LinkedIn's content understanding model, designed to generate high-quality post embeddings that power multiple downstream applications, including Feed ranking, Feed retrieval, out-of-network recommendations and immersive video experiences at a low dimensionality of 50. We also share how these embeddings are leveraged in a few downstream applications in a production setting for retrieval and ranking and demonstrate the utility of having embeddings that can capture semantics.

1.1 Key Contributions

- **Model Architecture for LinkedIn Post Embedding** – A multi-task fine-tuning approach for training LLMs on diverse datasets to generate post embeddings.
- **Offline Evaluation Methodology** - A semantic understanding metric to measure embedding quality.
- **Online Deployment** - Design for making embeddings accessible across LinkedIn's product surfaces.
- **Offline Results** - Offline results after multi-task fine-tuning including positive transfer among tasks and comparison against OpenAI embeddings on LinkedIn benchmarks.
- **Online Impact in Feed Ranking and Retrieval** - Evaluation of the real world effectiveness of LinkedIn post embeddings after deployment on the LinkedIn feed platform. We share practical examples of how embeddings can be integrated into ranking and retrieval models and the impact online through A/B testing. In this work, we share feed ranking, feed retrieval and video retrieval as examples.

2 Related Work

Pre-trained models such as BERT, RoBERTa, and more recently, models like GPT-3 [4] and E5 [11], have demonstrated remarkable performance improvements across a variety of Natural Language Processing (NLP) tasks. Recent studies have shown the effectiveness of fine-tuning models on multiple tasks to achieve better generalization and performance across all tasks. Aghajanyan et al. (2021) introduced MUPPET [1], which demonstrated that pre-fine-tuning on a diverse set of tasks could significantly improve the model's performance on individual tasks. Contrastive learning has emerged as a powerful technique for training embeddings by distinguishing between similar and dissimilar pairs. Reimers and Gurevych (2019) proposed Sentence-BERT, which trains both a siamese network architecture and a triplet architecture to generate embeddings for sentences, significantly improving performance on sentence similarity tasks [10]. Despite these advances, learning approaches can be sensitive to the quality of positive and negative pairs, and obtaining high-quality labeled data can be challenging. Models like XLM-R have shown that fine-tuning on multilingual data can lead to robust cross-lingual embeddings [5]. Greene et al. (2022) explored the performance of OpenAI's ADA embeddings, highlighting its performance on a multitude of tasks, underscoring its generalized nature [7]. However, these models often face limitations when applied directly to specific domains such as recommendation systems due to the linguistic variability between the general pre-trained embeddings and the specialized application domain. This requires additional fine-tuning to achieve optimal performance in domain-specific tasks. In our investigation, we encountered several challenges when trying to incorporate widely available architectures into production environments. These challenges included smaller context windows, limited linguistic variability in the topic ontology, and high embedding dimension size resulting in increased latency for production scale recommendation systems. In this paper, we build upon these foundational works by implementing a multi-task learning approach tailored to LinkedIn's unique content. By leveraging diverse semantic labeling tasks, we enhance the model's semantic understanding, improve multilingual support, and achieve competitive performance with significant compression. Our goal is to provide valuable insights into developing and deploying specialized embeddings in large-scale, real-world applications.

3 Vision for Training Platform

- **Accuracy**: the embeddings should accurately capture all of the relevant information about a post so that they can reliably be leveraged by downstream applications to make predictions about engagement or decisions about distribution.
- **Robustness**: the embedding should be able to handle all inputs relevant to the post, including text in any language used on LinkedIn. The training data did not have any filters based on language.
- **Timeliness**: to ensure the embeddings reflect the latest trends and patterns in the data, we should have the ability to retrain as often as needed
- **Extensibility**: Adding new sources of data or new tasks to the training pipeline needs to be standardized for any teams in LinkedIn to adopt or contribute data sources for fine-tuning.
- **Flexibility**: It should be easy to experiment with different underlying architectures for various base language models since this space is rapidly evolving.
- **Easy Deployability**: Seamless integration with required offline/nearline systems to ramp to production as quickly as possible.

4 Datasets

Dataset	Description
Interest	Derived from LinkedIn's topic tagging models, which classify posts into categories based on a structured ontology. A pair of posts is labeled positive if they share the same interest category.
Storyline	Editor-curated posts grouped by topics. Available in 50+ languages, this adds multilingual posts to the training data. These can be typically seen in the top right section of the LinkedIn feed in desktop
Hashtag	Uses post hashtags as soft labels. This is available in all languages on LinkedIn platform.
Search	Extracts query-post relevance pairs from LinkedIn's content search data.
Intent	Classifies posts based on intent (e.g., share advice, job seeking, motivation). **Used for evaluation to assess zero-shot generalization and not used for training the model.**

Table 1: Training and evaluation datasets overview

5 Modeling Architecture

In this section we first describe the single task training setup used for training content embeddings followed by the multi-task set up.

In our work, we have multiple tasks and multiple datasets. However, each data set is used for only one task. We did try other losses (margin maximization loss, prediction tasks) but none of them performed as well as the siamese architecture which we will illustrate in the subsections below.

5.1 Single Task Architecture

Figure 1 represents the siamese architecture which is a representation of a single task that was replicated across datasets for training a multi-task model [10]. We collect pairs of posts from the dataset (both positives and negatives). Positive examples are sampled in different ways for each dataset and the negative examples are two randomly sampled posts for every dataset (more details in section 4).

- Positive pairs (label = 1): P1 and P2 are topically related and should produce embeddings that have a high cosine similarity. Example: P1 and P2 are about bitcoin.
- Negative pairs (label = 0): P1 and P2 are NOT related and should produce embeddings that have a low cosine similarity. Example: P1 is about ML, P2 is about sports.

The label assigned is binary (1 or 0) and we apply a binary cross entropy loss between the label and the cosine similarity of the embeddings of the two posts.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\Big[y_i \cdot \log\left(\sigma\left(\cos(\mathbf{e}_1^{(i)}, \mathbf{e}_2^{(i)})\right)\right) + (1-y_i)\cdot\log\left(1-\sigma\left(\cos(\mathbf{e}_1^{(i)}, \mathbf{e}_2^{(i)})\right)\right)\Big] \tag{1}$$

Where:

- N is the number of pairs.
- $y_i \in \{0,1\}$ is the binary label for the i-th pair.
- $\mathbf{e}_1^{(i)}, \mathbf{e}_2^{(i)}$ are the embeddings for the i-th pair of posts.
- $\cos(\mathbf{e}_1, \mathbf{e}_2) = \frac{\mathbf{e}_1 \cdot \mathbf{e}_2}{\|\mathbf{e}_1\|\|\mathbf{e}_2\|}$ is the cosine similarity between two embeddings.
- $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function.

5.2 Multi-Task Architecture

We expand the single task setup to a multi-task training paradigm. The key idea of using a multi-task approach is that adding new datasets/tasks helps all the tasks being trained [1] versus training one model for every isolated task. The single task architecture described in the previous section is duplicated for multiple datasets (each dataset has its own independent task). We simultaneously train for several tasks with shared LLM parameters, allowing effective semantic representation to be efficiently learned within a single model. Each independent task tower computes its own loss. In our implementation **each task was an independent siamese task with BCE loss** similar to the set up in the single task architecture. With this approach we can fine-tune an LLM that has awareness of the semantics required for consumption by multiple downstream product teams. The parameters for the common LLM and the layer to reduce dimension are shared across all tasks. On completion of training, the task level heads are removed and the shared layers are used to infer the post embeddings. Figure 2 illustrates how the multi-task training is set up. The output of the [CLS] special token is employed as the pooling methodology to get the output from the LLM prior to dimension reduction.

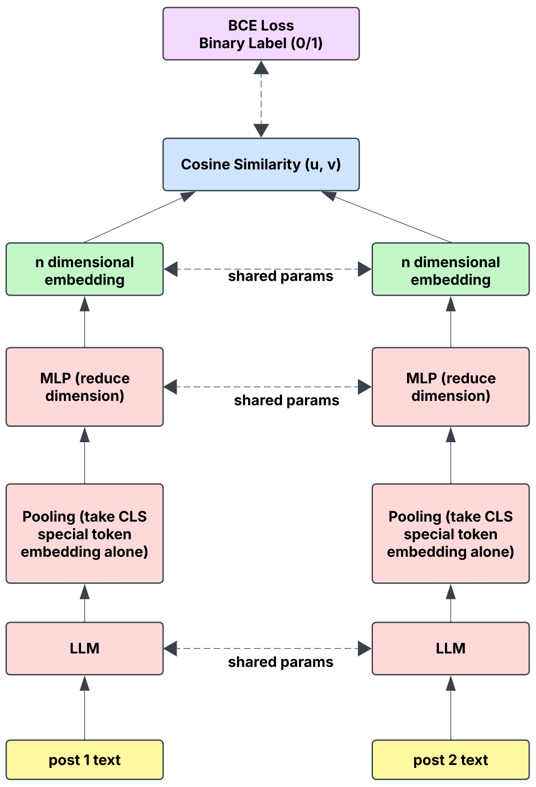

Figure 1: Architecture used for a single task

5.2.1 Task Heterogeneous Sampling. Similar to the implementation in the muppet paper [1], we sample data from all task data sources within one batch which helps with training stability and in gradient steps being better balanced across the tasks. Figure 3 illustrates the three steps below.

(1) For each dataset, randomly split it into the number of workers' splits.
(2) For each worker, load its corresponding split for all datasets.
(3) First batch data on each split, then randomly shuffle all the batches.

With this approach every batch consists of data from multiple datasets. The final loss employed was the average loss across all tasks. In the event of significant skew in dataset volume, a weighting term for each task could be added to help with the training. For this work, we did not add any weighting to the loss.

5.3 Implementation Details

We used 104M training samples coming from a combination of datasets described in section 4. We use a 6 layer multilingual BERT

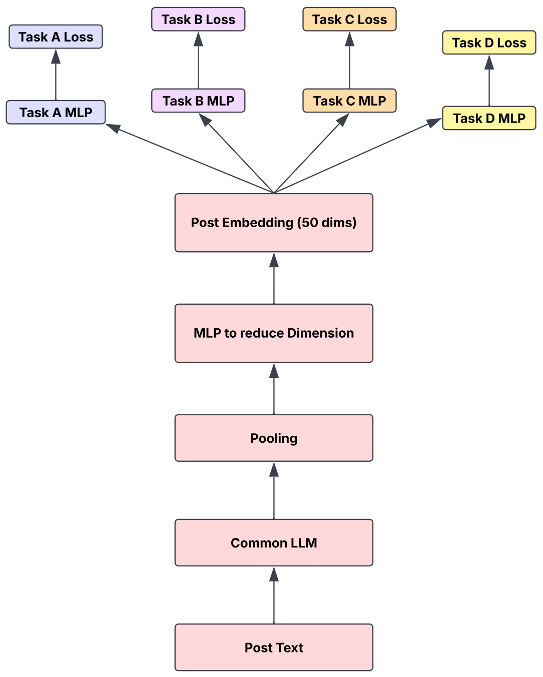

Figure 2: Multitask architecture

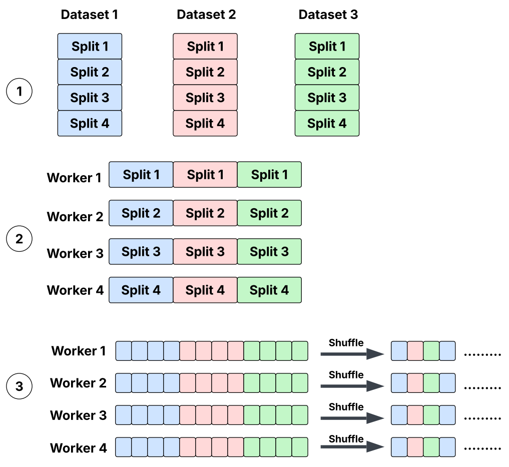

Figure 3: Task heterogeneous sampling with 3 datasets and 4 workers

(pre-trained on LinkedIn data using masked-language modeling) as the base model [6], with a total parameter size of 89M and vocabulary size of 135K. We use 1 worker and 6 GPUs for training. We use a per GPU batch size of 32 for siamese fine-tuning and shared embedding size of 50. We selected an embedding dimension of 50 after empirical experimentation. This dimension provides a balance between expressiveness and latency in large scale deployment. Higher dimensions achieved only marginally better offline accuracy while incurring higher inference cost and storage requirements. For task level parameters, each task has an MLP layer of size (50x100). We use a learning rate of 1e-6 for training. All experiments were conducted on a CentOS Linux server equipped with dual Intel® Xeon® Silver 4216 (Cascade Lake) CPUs (32 cores, 2.10 GHz), 64 GB of RAM, and an NVIDIA Tesla V100 SXM2 GPU with 32 GB memory, using CUDA Toolkit 11.7.

5.4 Member Embeddings from Post Embeddings

Numerous applications in LinkedIn require an embedding representation of LinkedIn members, and extracting this representation in an efficient way is essential. For example, any EBR (embedding-based retrieval) application needs both query (member) and item embeddings that are in the same space.

We made use of hierarchical clustering (Ward's method)[9] to get the representations of the member (medoids) based on member engagement in Feed and post embeddings of those engagements.

(1) Identify engagement history (e.g., like, comment, share, react)
(2) Join post embeddings to the corresponding history in engagement
(3) Run hierarchical clustering (Ward's method) to generate topK medoids for every member.

The top 'K' is based on an importance score for each cluster which is a combination of size of the cluster and freshness of items in the cluster [9].

6 Offline Evaluation

Training produces a model that captures post semantics that we evaluate across different downstream tasks. For instance, 2 posts on deep learning should be close in the embedding space. We built a simulation of EBR offline to evaluate the embeddings. We build a dataset of triplets containing anchor, positive and negative texts. The expectation is that for each anchor, the positive item is as close as possible and the negative item is as far as possible in the embedding space.

Anchor	Positive	Negatives
a_1	p_1	$\{n_{11}, n_{12}, \ldots, n_{1N}\}$
a_2	p_2	$\{n_{21}, n_{22}, \ldots, n_{2N}\}$
$\vdots$	$\vdots$	$\vdots$
a_M	p_M	$\{n_{M1}, n_{M2}, \ldots, n_{MN}\}$

Table 2: Anchor-Positive-Negative triplets

After training a candidate embedding model, we generate embeddings for all the text in the evaluation dataset, and then calculate the average fraction of triplets, where the distance between the anchor and positive instance is smaller than the anchor and negative instance. This serves as a good proxy for embedding based retrieval applications and is used as the offline evaluation metric for our content models.

AvgFracTripletsWherePosIsCloser: Fraction of triplets where the positive is closer to the anchor than the negative (larger is better):

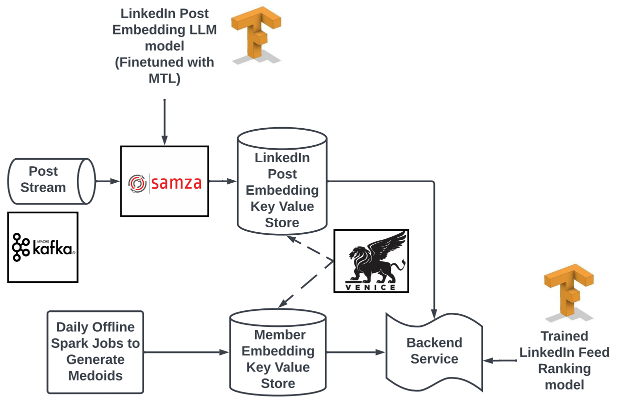
LinkedIn Post Embedding LLM model (Finetuned with MTL)
Post Stream
kafka
samza
LinkedIn Post Embedding Key Value Store
VENICE
Daily Offline Spark Jobs to Generate Medoids
Member Embedding Key Value Store
Backend Service
Trained LinkedIn Feed Ranking model

Figure 4: Online system for post embeddings

$$\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\begin{cases}1, & \text{if } \mathrm{dist}(a_i,p_i) < \mathrm{dist}(a_i,n_{ij})\\ 0, & \text{otherwise}\end{cases} \qquad (2)$$

7 Online Deployment

Figure 4 shows the high level overview of the online system we use in the Feed ranking model. All incoming posts that are created, are fed into a Samza job that computes embeddings using the trained LinkedIn post embedding model, and pushes it to a key-value store within 2 mins of post creation (typically this is done in a matter of seconds). The posts' key-value store is configured to store a fixed history of embeddings at all times on a rolling basis. Any backend service can fetch the embedding feature for scoring. Derived member embeddings (medoids), are pushed to a dedicated key-value store and this is an offline job which runs once a day.

8 Results

In subsection 8.1 to subsection 8.3 we will talk about offline results. To ensure fair evaluation, we report results on complete test data sets rather than sampled subsets. Multiple retraining runs produced only marginal differences in offline evaluation metrics, so we report representative results for clarity. For all of our online A/B tests, we report the statistical significance of our results in subsection 8.4. The performance improvements of our multi-task model are consistent across multiple datasets and tasks, reinforcing the robustness of our approach.

8.1 Fine-tuning an LLM on multiple tasks at once helps uplift performance in all tasks

T1, T2, and T3 correspond to models trained only on the independent datasets, and the last row is the LinkedIn Post Embedding model, trained in a multi-task fashion with all the datasets. E1, E2 and E3 are eval datasets built using the corresponding dataset mentioned in Table 3 (Interest, Storyline and Hashtag). The results demonstrate that our model trained on a combination of data from multiple semantic labeling tasks, shows a better overall performance across all tasks. The first 3 rows serve as an ablation

Model	E1	E2	E3
	(Interest)	(Story)	(Hashtag)
T1 (Interests)	0.88	0.86	0.79
T2 (Story)	0.76	0.93	0.85
T3 (Hashtag)	0.79	0.93	0.93
LinkedIn post embedding (MTL)	**0.89**	**0.95**	**0.93**

Table 3: Evaluation results across models using AvgFracTripletsWherePosIsCloser; E1-E3 correspond to different evaluation datasets

study showing the impact with the use of equivalent dataset only as opposed to MTL framework. Content search data was used for training but was not used for evaluation purposes, since there were no immediate plans to deploy these embeddings to the content search surface.

8.2 Zero Shot Capabilities Improvements

Model	E4 (Intent)
T4 (Intent)	0.69
LinkedIn Post Embedding (MTL)	**0.72**

Table 4: Evaluation results for intent understanding (E4) using AvgFracTripletsWherePosIsCloser

Table 4 demonstrates zero shot learning capabilities for the post embedding model. Although it is trained only on data from Interests, Search, Storylines and Hashtag datasets, it generalizes effectively to the Post Intent Dataset (E4). On Task E4, LinkedIn Post Embeddings outperforms the model fine-tuned solely on T4 (Post Intent training data).

8.3 Comparing performance with generalized OpenAI embeddings

E1 - Interest Dataset **E2** - Storyline Dataset **E3** - Hashtag Dataset

Model	Dim	E1	E2	E3
BERT-base	768	0.69	0.90	0.77
ADA_001	1024	0.66	0.95	0.82
ADA_002	1536	0.89	0.95	0.89
E5-base-v2	768	0.84	0.96	0.87
E5-multilingual-base	1024	0.81	0.96	0.87
LinkedIn Post Embedding	**50**	**0.89**	**0.95**	**0.93**

Table 5: Performance comparison across models (including ADA_002 [7]) using AvgFracTripletsWherePosIsCloser

The results in Table 5 show that compared to open-source models that generate generalized embeddings, we achieve comparable performance with up to 30x compression in embedding size for LinkedIn specific tasks.

8.4 Impact in Downstream Applications

All test results reported here are based on online A/B experiments run for at least one week, and all downstream application impacts are statistically significant with **p value less than 0.05**.

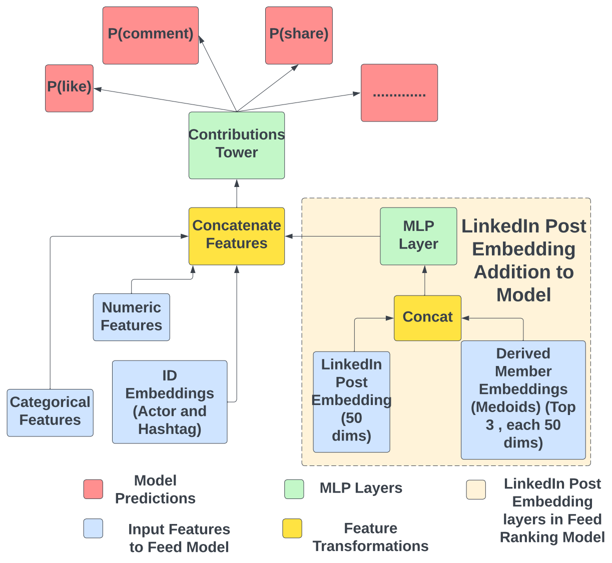
P(comment)
P(share)
P(like)
....
Contributions Tower
Concatenate Features
MLP Layer
LinkedIn Post Embedding Addition to Model
Concat
Numeric Features
Categorical Features
ID Embeddings (Actor and Hashtag)
LinkedIn Post Embedding (50 dims)
Derived Member Embeddings (Medoids) (Top 3 , each 50 dims)
Model Predictions
MLP Layers
LinkedIn Post Embedding layers in Feed Ranking Model
Input Features to Feed Model
Feature Transformations

Figure 5: Feed ranking model architecture with LinkedIn post embeddings

8.4.1 Feed Ranking: The Feed ranking model is the final ranking layer, which takes in inputs from multiple first pass rankers (examples: followed content, jobs content, suggested content, etc.), and outputs the final ranked list. The current Feed ranking model is a large personalized model which includes ID features, numeric features and categorical features [3]. Figure 5 shows how the embeddings were integrated into the main Feed ranking model.

After adding LinkedIn Post Embeddings to the model, we were able to achieve an increase of **0.1% in the number of user sessions** on LinkedIn ($p < 0.001$). This metric indicates that more members found the feed more relevant and chose to come back more often. We achieved an increase of **0.21% in the number of daily unique professional interactions** by our members ($p < 0.0001$) and **revenue** was up by **0.42%** ($p < 0.001$) in online A/B tests. While these lifts may appear modest, at LinkedIn's scale, this translates to millions of additional positive member interactions daily, representing a significant business impact.

8.4.2 Feed Retrieval: The Feed model has a retrieval layer for fetching the best candidates for ranking from the corpus of content created by a member's connections. This is one of the sources that feeds into the final ranking model along with out-of-network posts, videos, ads, jobs and others. In this layer, we filter down top 500 most relevant connected content for the user from a corpus which could range up to millions depending on connection size of a member. See Figure 6 for the architecture of the retrieval model after addition of LinkedIn Post Embeddings. The model resulted in a **+0.37% increase in daily unique members who had an active engagement in the Feed** ($p < 0.001$), and a **0.05% decrease in Feed skips, indicating greater relevance** ($p < 0.0001$).

8.4.3 Feed Video Recommendation: LinkedIn has a video recommendation experience across multiple product surfaces like Video

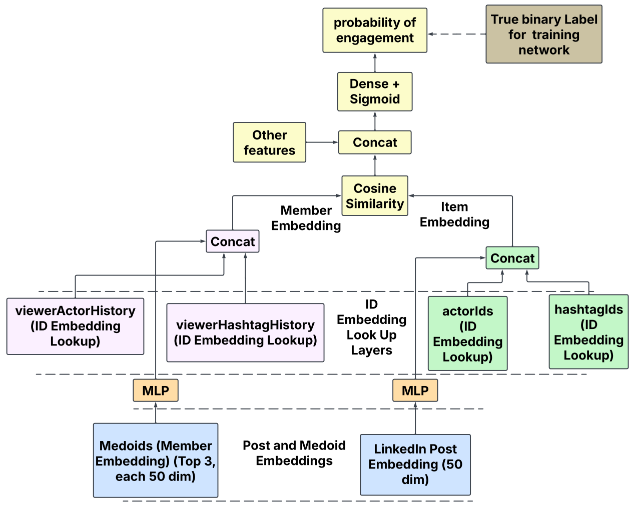
probability of engagement
True binary Label for training network
Dense + Sigmoid
Other features
Concat
Cosine Similarity
Member Embedding
Item Embedding
Concat
Concat
viewerActorHistory (ID Embedding Lookup)
viewerHashtagHistory (ID Embedding Lookup)
ID Embedding Look Up Layers
actorIds (ID Embedding Lookup)
hashtagIds (ID Embedding Lookup)
MLP
MLP
Medoids (Member Embedding) (Top 3, each 50 dim)
Post and Medoid Embeddings
LinkedIn Post Embedding (50 dim)

Figure 6: Feed retrieval model architecture with LinkedIn post embeddings

Tab, Video Chaining and Video Carousel where professional content in the format of short videos are available for our members. We added post embeddings based on video transcript information to the video retrieval layer (detailed architecture in Figure 7) and we achieved an improvement of **+10.46%** in **Total Watch Time** ($p < 0.0001$) and **+1.74%** in **DAU** ($p < 0.0001$) for the video tab surface. The LinkedIn Post Embedding model was not retrained, and was simply used for inference using the video transcript text as input to generate embeddings. This further validates the ability of the embeddings to easily scale across multiple product surfaces with impact.

9 Conclusion

In this paper, we presented the fine-tuning of a BERT-based language model to generate high quality post embeddings, that have been widely adopted across LinkedIn. We demonstrated how these embeddings enable the derivation of member representations using hierarchical clustering (Ward's method), and showed that training on a diverse set of semantic labeling tasks led to consistent performance improvements through positive transfer. The generalizability of the model was further validated through its zero-shot learning capabilities on an unseen task. We also compared the model's performance against OpenAI's generalized embeddings, highlighting its superior effectiveness for LinkedIn specific applications achieved at significantly lower dimensionality. Furthermore, we outlined several real world product use cases where these embeddings were deployed, resulting in measurable online impact through A/B testing. Looking ahead, our goal is to integrate larger foundational language models into our fine-tuning framework, and expand our training datasets by partnering with additional LinkedIn product teams. This ongoing effort towards platformization not only enhances the quality and scalability of post embeddings, but also incentivizes the creation of new, reusable datasets that can benefit multiple teams across the company.

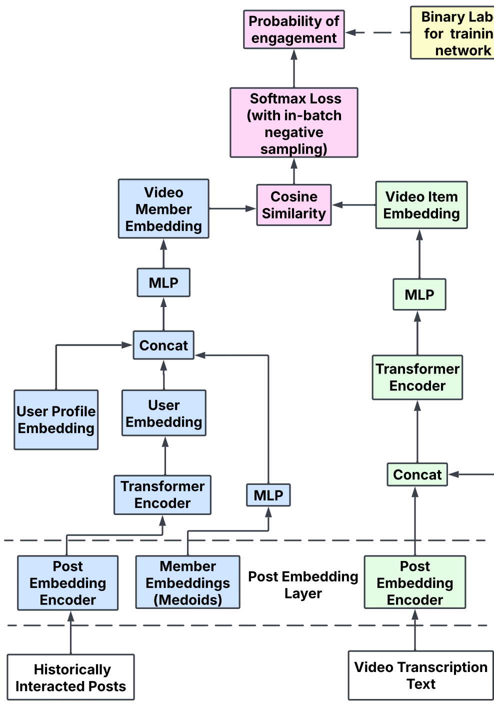

Figure 7: Video retrieval model architecture with LinkedIn post embeddings

Limitations

Our embeddings have been battle-tested in production for over two years and remain competitive, although advances in foundational and multimodal models may eventually surpass this approach. Our framework is designed to incorporate such improvements. We chose binary cross-entropy loss for efficiency: while triplet loss and InfoNCE are standard, triplet loss required three parallel inferences per update, increasing GPU memory costs and reducing batch sizes, which degraded performance. BCE provided a more practical trade-off between efficiency and downstream accuracy. Our evaluation metric (*AvgFracTripletsWherePosIsCloser*) aligns with our semantic understanding objective, although alternative ranking or retrieval based offline metrics could provide complementary views of embedding quality. Finally, embedding performance is sensitive to how positive and negative pairs are sampled; although we adopted a broad, multi-task strategy, more sophisticated pair generation may further improve generalization. We have initiated efforts to develop multimodal post embeddings that integrate both visual and textual information, since the current embeddings only operate on the text present in a post.

References

[1] Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038* (2021).

[2] Fedor Borisyuk, Shihai He, Yunbo Ouyang, Morteza Ramezani, Peng Du, Xiaochen Hou, Chengming Jiang, Nitin Pasumarthy, Priya Bannur, Birjodh Tiwana, et al. 2024. LiGNN: Graph Neural Networks at LinkedIn. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4793–4803.

[3] Fedor Borisyuk, Mingzhou Zhou, Qingquan Song, Siyu Zhu, Birjodh Tiwana, Ganesh Parameswaran, Siddharth Dangi, Lars Hertel, Qiang Charles Xiao, Xiaochen Hou, et al. 2024. LiRank: Industrial Large Scale Ranking Models at LinkedIn. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4804–4815.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116* (2019).

[6] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Ryan Greene, Ted Sanders, Lilian Weng, and Arvind Neelakantan. 2022. New and improved embedding model. *OpenAI Blog. Available online: https://openai.com/blog/new-and-improved-embedding-model (accessed on 28 November 2023)* (2022).

[8] Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* 364 (2019).

[9] Aditya Pal, Chantat Eksombatchai, Yitong Zhou, Bo Zhao, Charles Rosenberg, and Jure Leskovec. 2020. Pinnersage: Multi-modal user embedding framework for recommendations at pinterest. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2311–2320.

[10] N Reimers. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv preprint arXiv:1908.10084* (2019).

[11] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533* (2022).

[12] Spyros Zoupanos, Stratis Kolovos, Athanasios Kanavos, Orestis Papadimitriou, and Manolis Maragoudakis. 2022. Efficient comparison of sentence embeddings. In *Proceedings of the 12th Hellenic Conference on Artificial Intelligence*. 1–6.