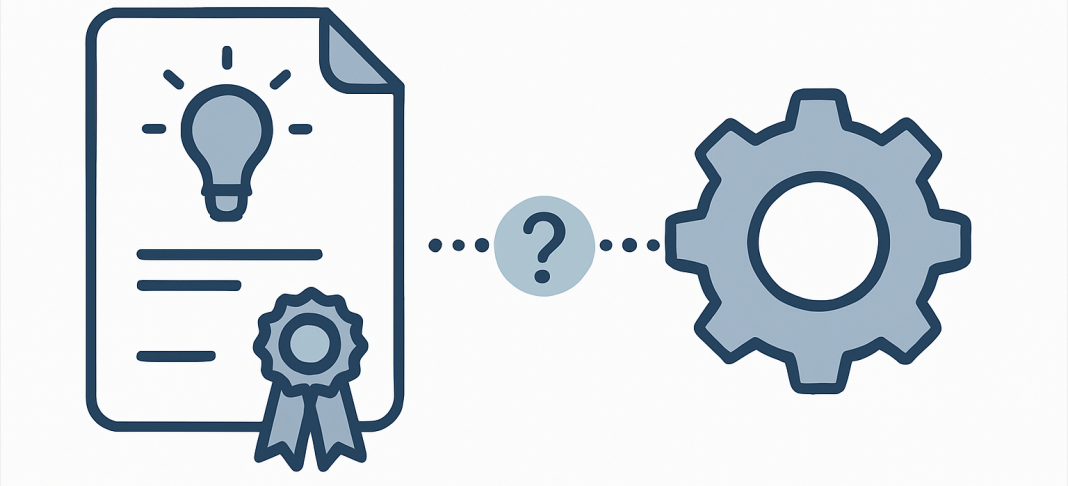
Product-Patent linkage prediction

Mehdi El Bouari
mehdi.elbouari@epfl.ch
EPFL
Lausanne, Switzerland

Gaétan de Rassenfosse
gaetan.derassenfosse@epfl.ch
EPFL
Lausanne, Switzerland



Abstract

Linking commercial products with relevant patents is a crucial step in intellectual property analysis, competitive intelligence, and innovation tracking. However, establishing such connections is challenging due to sparse annotations and the semantic gap between product descriptions and patent documents. In this project, we present a machine learning pipeline to automatically associate products with related patents using a combination of contrastive learning and adversarial hard negative mining. We construct a bipartite graph where nodes represent products and patents, and edges denote known associations, and use an automated data extraction pipeline to extract textual data pertaining to the products and patents, coupled to an LLM based compression pass to generate dense and rich textual features. Our model learns domain-specific embeddings for both products and patents by optimizing a contrastive loss, while an adversary is trained to generate difficult negative samples to improve robustness. Experimental results demonstrate the effectiveness of our approach in capturing nuanced semantic relationships and improving link prediction performance across sparse product-patent datasets.

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

CCS Concepts

• **Information systems** → Language models; • **Computing methodologies** → **Information extraction**; *Natural language generation*; **Adversarial learning**; **Learning to rank**; • **Applied computing** → **Document searching**; **Document analysis**.

Keywords

Product-Patent Linkage, Contrastive Learning, Adversarial Training, Data Engineering

ACM Reference Format:
Mehdi El Bouari and Gaétan de Rassenfosse. 2018. Product-Patent linkage prediction. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In the rapidly evolving landscape of innovation, connecting commercial products to relevant patents is a task of growing importance. Such linkages enable enterprises to monitor the intellectual property (IP) space, identify competitive threats, and assess the novelty or infringement risk of new offerings. Despite their value, these associations are rarely available at scale, and manual curation is time-consuming and error-prone. Products and patents are described in different styles, vocabularies, and levels of abstraction, making the task particularly challenging for automated systems.

To address this problem, we explore a learning-based approach to infer links between products and patents from textual data. Our solution is centered around a dual-encoder architecture that embeds product and patent descriptions into a shared vector space, optimized using a contrastive loss function. To improve discriminative

power and robustness, we incorporate an adversarial component that generates hard negative samples, encouraging the model to focus on subtle yet meaningful distinctions.

To ensure fair and realistic evaluation, we model our data as a bipartite graph and develop a custom graph splitting strategy that maintains balanced node and edge distributions while isolating high-impact components for targeted testing. Our findings show that this text-based embedding generation coupled with adversarial hard negative mining and graph-aware evaluation—offers a scalable and effective solution for product-patent linkage.

2 Data

One of the major challenges in building a product-patent linkage prediction system is the lack of publicly available, well-structured datasets that connect commercial products to their corresponding patents. This gap stems from the fact that companies are not legally required to disclose such linkages at the time of commercialization. While laws historically required products to be marked as "Patented" along with the relevant patent numbers, this practice—known as actual marking or physical marking—has several limitations in modern industrial contexts.

Physically marking products with patent numbers can be impractical for companies with large or fast-growing patent portfolios. Packaging constraints, frequent product updates, and the need to manage variations across product lines make maintaining accurate physical markings a costly and error-prone process. For instance, each time a new patent is granted for a product, manufacturers would need to update labels, molds, or packaging—introducing logistical and financial burdens.

To alleviate these challenges, the U.S. Congress amended patent laws to allow virtual marking. In this system, products are labeled with the word "patent" (or "pat.") followed by a URL that hosts a list of applicable patent numbers. This approach simplifies legal compliance and reduces manufacturing overhead, especially for small products or for companies managing large or rapidly changing patent portfolios.

In this project, we leverage the advent of Virtual Patent Marking (VPM) to identify reliable data sources for constructing product-patent pairs. Specifically, we focus on Honeywell International Inc., a multinational conglomerate operating across aerospace, industrial automation, building systems, and energy sectors. Honeywell maintains an extensive catalog of commercial products and patented technologies. Recently, the company published a Virtual Patent Marking (VPM) page that explicitly links many of its products to the relevant patents protecting them. This publicly available resource provides a rare and valuable dataset of real-world product-patent linkages.

We curated our dataset by extracting structured product and patent information from Honeywell's VPM page. This dataset serves as the foundation for training and evaluating our product-patent linkage prediction model. In the following sections, we detail our methodology for extracting accurate and meaningful textual representations of both products and patents.

2.1 Patent Data

To generate the textual representations of patents, we utilize Google's Public Patent Datasets, accessible via BigQuery, which provide comprehensive and structured metadata for patents hosted on the Google Patents platform.

For each patent linked in the Honeywell VPM dataset, we extract the following key textual fields:

- Title
- Abstract
- Claims
- Description

These fields collectively capture both high-level and detailed technical aspects of the invention. To ensure the information is current, we filter the dataset to retain only the most recent version of each patent entry.

The concatenation of these fields effectively reconstructs the full textual content available on a typical Google Patents page. However, the combined length often exceeds the input limit of our text encoder. To address this, we apply an LLM-based summarization pipeline that compresses the full text into a highly information-dense summary. This allows us to preserve the most salient technical and legal information while keeping the input within the encoder's maximum length constraints.

2.2 Product Data

One of the most challenging aspects of this project is extracting accurate and relevant textual descriptions for each product listed on the Honeywell VPM page. Unlike patents, which are indexed in centralized and structured databases like Google Patents, product information is scattered across heterogeneous and inconsistent sources, making large-scale extraction particularly difficult. We encountered several major obstacles:

(1) **Heterogeneous Data Sources:** Product data had to be collected from a wide range of websites. The Honeywell VPM page often lacks direct links to product pages, and many products were originally commercialized by companies that were later acquired by Honeywell (e.g., Intermec). As a result, information is distributed across legacy websites and archived catalogs, leading to significant variability in format, structure, and content quality.
(2) **Data Scarcity:** Many products listed in the VPM dataset are either discontinued or replaced by newer models. Official information about these older products is often removed from Honeywell's current websites, forcing us to rely on third-party sources such as distributors, documentation archives, or cached web pages. Furthermore, some products—especially those used in military or aerospace applications—are intentionally documented with limited public detail to preserve trade secrets or comply with regulatory restrictions.
(3) **Unclear Product Designations:** The product identifiers listed on the VPM page are often ambiguous or inconsistent. In some cases, the same product name refers to multiple distinct models or product lines. Additionally, while the VPM page may list specific variants of a product, public documentation typically focuses on broader, generic models. This

mismatch reduces the specificity of the textual information available for many products.

(4) **Extraneous or Irrelevant Data:** Even when product information is available, much of it pertains to commercial or logistical attributes (e.g., packaging dimensions, compliance certifications), which are not directly relevant to the task of patent linkage. Conversely, information about innovative technologies or internal components—critical for understanding the product's patent coverage—is often sparse, vague, or intentionally omitted.

Despite these challenges, we develop a data extraction pipeline to automatically extract and uniformlly format the product data, before we manually inspect and complement the pipeline's results. In the following section we present the different steps of or product data extraction pipeline :

2.2.1 Website Gathering. The first step in our product data extraction pipeline involved identifying web sources that contain relevant information for each product. Given the heterogeneity and fragmentation of product data, this step is crucial to ensure both coverage and quality of the extracted content.

To retrieve relevant webpages, we issued a set of structured search queries to Google for each product in the VPM dataset. Specifically, we used the following four query templates:

- `Honeywell product`
- `Honeywell product datasheet`
- `Honeywell product technical specifications`
- `Honeywell product user manual`

Each of these queries was designed to maximize the likelihood of retrieving rich and informative content. While the general query helps locate official or commercial pages, the datasheet and specification queries tend to surface technical documentation, and the user manual query often links to detailed usage information or maintenance instructions.

For each query, we automatically extracted the top-ranked Google search result and recorded the corresponding URL. This process yielded up to four candidate webpages per product, which were then passed on to the next stage of our pipeline.

2.2.2 Website Filtering. To ensure the accuracy and relevance of our extracted product information, we applied a multi-stage filtering process aimed at minimizing false positives. This step is especially important for products with ambiguous names or limited public documentation, which often lead to incorrect or low-quality search results.

The filtering process consists of two main phases:

(1) **Deduplication and Manual Heuristics:** We begin by removing duplicate URLs for each product to reduce redundancy and avoid unnecessary processing. Next, we manually reviewed the most frequently returned domains across products and excluded sources that consistently produced irrelevant or low-information content. For instance, some obscure product queries led to the Honeywell VPM page itself being returned as a top result, which we filtered out as it does not contain descriptive product information.

(2) **LLM-Based URL Relevance Filter:** In the second phase, we employed a lightweight LLM-based classifier that evaluates the relevance of a URL given a product name. The model operates using a simple heuristic: it checks whether the product name (or a close variant) appears in the URL path or domain. While this rule-based approach is relatively aggressive and may exclude some true positives, it significantly reduces irrelevant links. Given our ability to later manually augment data where necessary, we prioritized precision over recall in this stage to avoid polluting the dataset with false matches.

This filtering pipeline strikes a balance between automated efficiency and manual curation, providing a cleaner and more trustworthy set of candidate webpages for downstream content extraction.

2.2.3 Data Extraction. Once we obtained our filtered set of URLs for each product, we proceeded to scrape the content of the corresponding webpages and extract relevant product information. This phase required our pipeline to handle a wide variety of web sources, including HTML pages from e-commerce sites, PDF datasheets, and occasionally encrypted or dynamically loaded content.

Given the unstructured and often lengthy nature of the retrieved content, we segmented the raw text into manageable chunks. Each chunk, along with the associated product name, was then passed to a lightweight large language model (LLM) designed to identify and extract concise, relevant information.

For this purpose, we utilized Google's Gemma3[6] (27B) model family. These models offer near state-of-the-art performance on a variety of language understanding benchmarks while maintaining a significantly smaller parameter count, making them well-suited for efficient inference on commodity hardware.

We chose to represent the extracted product information in the form of bullet points. This structured format allows us to aggregate insights across multiple sources and content chunks, yielding a compact list of factual information for each product.

2.2.4 Data Formatting. The final step in our product data pipeline involves processing the extracted bullet points for each product, removing redundancy, and formatting the remaining content into a compact natural language description.

We again employ a lightweight LLM to filter out duplicate, irrelevant, or contradictory bullet points to ensure the integrity and clarity of the final representation and convert the cleaned bullet points into a coherent paragraph-style description for each product.

This transformation serves two purposes: (1) it produces a uniform, structured representation of product information across highly heterogeneous sources, and (2) it facilitates downstream usage by providing model-ready textual inputs for the product-patent linkage prediction task.

By standardizing the input format, we ensure that the model receives high-quality, semantically dense descriptions, which are essential for learning meaningful associations between products and their corresponding patents.

2.2.5 Manual Complementation. As a final step in our data preparation process, we manually reviewed and complemented the extracted product descriptions. This stage is critical for products

where the automated pipeline failed to retrieve meaningful or accurate information.

Failures typically occurred under the following circumstances:

(1) **Ambiguous product names:** Confusing or generic product designations often led to irrelevant or unrelated search results.
(2) **Discontinued products:** For older or retired models, official information was often scarce or no longer available online.
(3) **Access-restricted websites:** Some relevant sources were encrypted, region-locked, or otherwise inaccessible to automated scraping tools.

In such cases, we manually searched for and incorporated missing or corrected information using alternative sources, including archived documentation, technical forums, and product brochures. Additionally, we evaluated the accuracy of the automatically generated product descriptions on a representative subset of the dataset. Descriptions that were found to be incomplete, vague, or factually incorrect were revised accordingly.

The overall performance of this data extraction and manual correction pipeline is detailed in the Results section.

3 Contrastive learning framework

In this section we describe different elements of the contrastive learning training pipeline of our product-patent link prediction model.

3.1 Data Splitting Strategy

The first key component of our framework is the strategy used to split the dataset into training, validation, and test sets. A rigorous and fair evaluation of our model requires that the validation and test sets be fully isolated from the training set. However, due to the structure and sparsity of our data, this splitting process presents several non-trivial challenges.

Our cleaned dataset consists of 6,269 nodes in total—1,826 products and 4,443 patents. In a fully connected bipartite graph between these two sets, there would be up to 8,112,918 possible edges. However, our actual dataset contains only 20,192 observed edges, accounting for a mere 0.2% of all possible connections. This extreme sparsity significantly complicates learning and generalization, as the model is exposed to only a tiny fraction of the potential space of product-patent relationships.

In addition to sparsity, the node degree distribution is highly skewed: some nodes are densely connected, while others are nearly isolated. This heterogeneity makes it difficult to split the graph while maintaining consistent coverage and proportions of nodes and edges across splits.

Furthermore, the graph is composed of many disconnected components. These components often correspond to distinct domains or sectors—for example, all nodes related to chemical products may form a tightly-knit subgraph. If such a domain-specific cluster is assigned entirely to either the training or test set, the evaluation will be biased and fail to reflect the model's generalization ability across domains.

To address these challenges, we develop a custom graph-based splitting strategy aimed at preserving structural diversity and achieving balanced node and edge distributions across the training and test sets.

We begin by analyzing the disconnected components of the bipartite product-patent graph, examining their density as well as node and edge counts. The results of this analysis are shown in Figure 1.

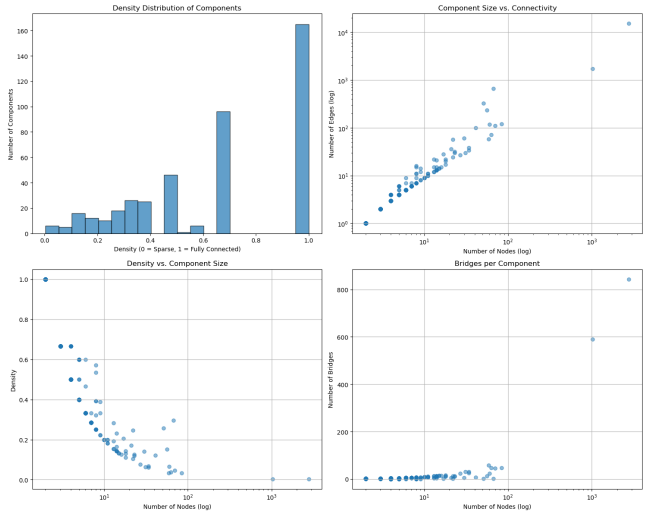


Figure 1: Component sparsity and statistics

As shown in the figure, two outlier components dominate the graph structure, containing approximately 61% of all nodes and 83% of the total edges. To ensure the test set is representative of the training data distribution, we apply a bridge-cutting strategy to these two large components. Specifically, we iterate over candidate bridges (edges whose removal increases the number of connected components) and evaluate how their deletion affects the resulting node and edge proportions. The goal is to split each large component into two subgraphs such that one of them can serve as a test set with statistics close to our target distribution. The remaining smaller components are then assigned directly to either the training or test set without modification, minimizing information loss.

To guide this process, we define a loss function based on the L2 distance between the actual and target counts of product nodes, patent nodes, and edges in the test split. This loss function helps the algorithm decide whether to assign each component to the test or training set in a way that minimizes deviation from the desired proportions.

The algorithm runs in two passes. The first is a top-down pass, where we start by splitting the largest components using the bridge-cutting strategy and then assign the remaining smaller components in a way that minimizes the global loss. The second is a bottom-up pass, where we reverse the order—assigning the smaller components first and then selecting the optimal bridge split for the large components. Both passes yield candidate splits, and we retain the one that achieves the lowest overall loss.

This strategy allows us to navigate the heterogeneous and sparse structure of the graph, generating test (or validation) splits that are both statistically balanced and structurally diverse, enabling a more robust and unbiased evaluation of our model.

3.2 Encoder Architecture

Our model is based on a contrastive learning framework employing a dual-encoder architecture, designed to learn a shared representation space for products and patents.

Each encoder is built upon a pretrained PaECTER model[1], a version of PatentBERT[4] finetuned to produce general patent embeddings. This initial encoder is frozen and used to processes the summarized textual descriptions of products and patents and generate initial embeddings for every token.

To transform the token-level embeddings produced by PaECTER into a single vector representation, we apply a multi-head attention pooling mechanism with learnable parameters. This pooling step enables the model to attend to the most informative parts of the input text, effectively learning which segments are most relevant for the linkage task.

The pooled representation is then passed through a series of feed-forward layers equipped with residual connections, layer normalization, and dropout. These layers enhance the model's expressive power while improving generalization and training stability.

We instantiate two separate encoder networks with this architecture:

- A **product encoder**, responsible for encoding product descriptions.
- A **patent encoder**, responsible for encoding patent texts.

While both encoders share the same underlying architecture, they are trained independently to capture domain-specific nuances of product and patent texts, respectively. This dual-encoder setup allows us to compute meaningful similarity scores between product and patent embeddings for contrastive learning.

3.3 Supervised Contrastive Loss

For our loss function, We implement a supervised contrastive loss function[2] that operates on product-patent pairs.

Let $\mathbf{P} \in \mathbb{R}^{n \times d}$ be the normalized product embeddings matrix and $\mathbf{Q} \in \mathbb{R}^{m \times d}$ be the normalized patent embeddings matrix, where n is the number of products, m is the number of patents, and d is the embedding dimension.

The similarity matrix is computed as:

$$\mathbf{S} = \mathbf{P}\mathbf{Q}^T / \tau \in \mathbb{R}^{n \times m} \tag{1}$$

where τ is the temperature scaling hyperparameter and S_{ij} represents the scaled cosine similarity between product i and patent j.

Let $\mathcal{P}_{i\cdot} \subseteq \{1, \ldots, m\}$ be the set of patents related to product i, and $\mathcal{N}_{i\cdot} \subseteq \{1, \ldots, m\}$ be the set of patents not related to product i.

Similarly let $\mathcal{P}_{\cdot j} \subseteq \{1, \ldots, n\}$ be the set products related to patent j, and $\mathcal{N}_{\cdot j} \subseteq \{1, \ldots, n\}$ be the set of products not related to patent j.

The supervised contrastive loss consists of two terms to ensure bi-directionality across patents and products :

3.3.1 Product Loss. For each product i that has at least one positive and negative patent in the batch:

$$\mathcal{L}_{\text{prod}}^{(i)} = -\frac{1}{|\mathcal{P}_{i\cdot}|} \sum_{j \in \mathcal{P}_{i\cdot}} \log \frac{\exp(S_{ij})}{\sum_{k \in \mathcal{N}_{i\cdot}} \exp(S_{ik})} \tag{2}$$

3.3.2 Patent Loss. For each patent j that has at least one positive product:

$$\mathcal{L}_{\text{pat}}^{(j)} = -\frac{1}{|\mathcal{P}_{\cdot j}|} \sum_{i \in \mathcal{P}_{\cdot j}} \log \frac{\exp(S_{ij})}{\sum_{k \in \mathcal{N}_{\cdot j}} \exp(S_{kj})} \tag{3}$$

3.3.3 Total Loss. The total supervised contrastive loss is then computed as the average of product and patent losses :

$$\mathcal{L} = \frac{1}{2}\left(\frac{1}{n}\sum_{i=0}^{n} \mathcal{L}_{\text{prod}}^{(i)} + \frac{1}{m}\sum_{j=0}^{m} \mathcal{L}_{\text{pat}}^{(j)}\right) \tag{4}$$

This loss function is particularly well-suited for our task, as it enables the model to bring positive product-patent pairs closer in the embedding space while simultaneously pushing apart representations of unrelated pairs. By leveraging label supervision, it encourages the formation of well-separated and semantically meaningful clusters in the representation space, which is essential for accurate link prediction between products and patents.

A major advantage of this loss is its ability to generalize to an arbitrary number of positive examples. Unlike traditional contrastive loss formulations that rely on fixed pairs or triplets, the supervised contrastive loss allows all samples sharing the same label (i.e., all valid product-patent matches) to contribute to the positive term in the loss. This leads to a stronger, more robust alignment of semantically similar representations, particularly in large batches, where multiple positives are likely to be present. This property improves the model's ability to form coherent clusters of related products and patents, thus enhancing retrieval and classification accuracy.

Furthermore, the loss retains the summation over all negative samples in the denominator, which significantly boosts its contrastive power. The presence of a large and diverse set of negative examples—especially in high-batch training—encourages the model to better distinguish between relevant and irrelevant pairs. This helps reduce false positives, which is particularly important in our context where many patents and products share overlapping terminology but refer to distinct technologies.

Another critical benefit of the supervised contrastive loss is its intrinsic ability to perform hard positive and hard negative mining implicitly. Thanks to the structure of its gradient, the loss automatically emphasizes examples that are more difficult to contrast—such as similar but incorrect pairs or distant positives—without requiring explicit mining strategies. This is especially valuable in our setting, where noisy data and fine-grained distinctions between products and patents are common, and where the sparsity of the dataset introduces a large set of potential negatives.

Additionally we also implement dynamic gradient clipping by using zclip[3], to dynamically keep track of gradient norms and variations and learn clipping

Taken together, these properties make the supervised contrastive loss a powerful and robust choice for our product-patent linking task, offering a strong combination of scalability, representational quality, and ease of implementation.

3.4 Adversarial Training

As previously discussed, our patent-product graph is extremely sparse, making hard negative mining a critical challenge for effective training. To address this, we implement a novel hard negative mining strategy inspired by reinforcement learning, specifically policy gradient methods.

We leverage the fact that our product and patent spaces are discrete and fixed during training. This allows us to train two separate adversarial policies:

(1) **Product adversary:** Takes a patent embedding as input and generates a probability distribution over the discrete set of training products.
(2) **Patent adversary:** Takes a product embedding as input and generates a probability distribution over the discrete set of training patents.

Each of these adversaries learns to assign higher probabilities to "hard negatives"—i.e., samples that are difficult for the encoder to distinguish from positives. The predicted probability distributions are then masked to exclude positive pairs, entries already present in the batch, and duplicates among selected negatives. We sample from these masked distributions without replacement to generate a diverse set of high-quality hard negatives.

When combined with our supervised contrastive loss—which jointly attends to all positives and negatives within a batch—this adversarial mining strategy enables more thorough exploration of the dataset and significantly improves the model's generalization performance.

The adversaries are trained using a policy gradient loss defined as:

$$\mathcal{L}^{i}_{adv} = -\frac{1}{|\mathcal{H}_i|} \sum_{j \in \mathcal{H}_i} R_i(j) \times \log(\pi_i(j))$$

where $\mathcal{H}_i$ is the set of selected hard negatives for a given query (product or patent) i, $\pi_i(j)$ is the predicted probability of selecting negative j, and $R_i(j)$ is a reward signal that reflects the quality of the sampled negative.

Since our contrastive learning objective encourages the similarity of positives to be higher than that of negatives, a natural reward for the adversary is the similarity between the query and the sampled negative:

$$R_i(j) = S_{ij}$$

This similarity-based reward is detached from the computation graph to prevent gradient flow through the reward and to maintain stable training.

A central challenge in training policy gradient algorithms[5] is managing the exploration-exploitation tradeoff. If the policy converges too quickly to high-reward actions (exploitation), it may fail to adequately explore the full action space and overlook better alternatives. Conversely, excessive exploration without convergence can lead to unstable or inefficient learning. Traditional reinforcement learning frameworks address this by using stochastic policies and encouraging early exploration before gradually shifting toward exploitation as the model learns.

In our setting, we benefit from a natural form of structured exploration due to how we sample negatives. Specifically, the adversarial policies generate a probability distribution over the entire set of potential negatives, and we sample multiple negatives without replacement at each training step. This ensures that a diverse subset of the action space is explored in every forward pass. Consequently, the model can evaluate and learn from a variety of candidate negatives simultaneously, which facilitates efficient learning of the reward landscape without requiring explicit exploration mechanisms.

To further stabilize training and reduce variance in the gradient estimates, we introduce a baseline into the policy gradient loss. The modified loss becomes:

$$\mathcal{L}^{i}_{adv} = -\frac{1}{|\mathcal{H}_i|} \sum_{j \in \mathcal{H}_i} (R_i(j) - b) \times \log(\pi_i(j))$$

where b is a baseline value used to normalize the reward signal. In our implementation, we set b as the mean similarity score among the sampled negatives in the batch. This baseline acts as a reference point, so that only deviations from the average reward contribute to the gradient update. Negatives that are more similar than average receive positive reinforcement, while those that are less similar are penalized. This variance reduction technique is standard in policy gradient methods and is particularly effective in our scenario, where reward signals based on similarity scores can vary significantly. Incorporating the baseline leads to more stable updates, reduces noise in training, and accelerates convergence toward useful hard negatives.

To avoid destabilizing the model in the early stages of training, we begin with a warm-up phase where no hard negatives are introduced during the first epoch. This allows the encoders to build an initial representation of the data and learn the general structure of the embedding space without interference from adversarial sampling. In the second epoch, we introduce 5 hard negatives (both patents and products) per training sample, enabling the adversaries to begin exploring the action space and learning which negative samples are more informative. Finally, in the third epoch, we increase the number of hard negatives to 10 per sample, allowing the model to fully leverage the adversaries' learned policies and focus on exploiting the most challenging examples to further refine the embedding space.

Figure 2 illustrates the evolution of the policies generated by the adversaries for a select product and patent throughout training, and we can clearly see the policy converging towards a couple of negatives, showcasing the convergence of the adversary.

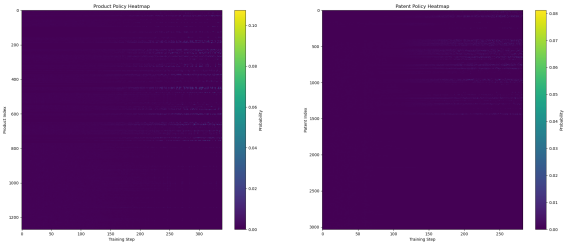


Figure 2: Adversary policies throughout training

4 Results

In this section, we present the outcomes of our training pipeline and evaluate the performance of our model on both the training and test sets. We use several metrics to provide a comprehensive picture of model behavior, including loss curves, adversarial reward progression, Precision@k, Mean Reciprocal Rank (MRR), Mean Average Precision (MAP), and ROC/AUC and Precision-Recall curves.

4.1 Training set

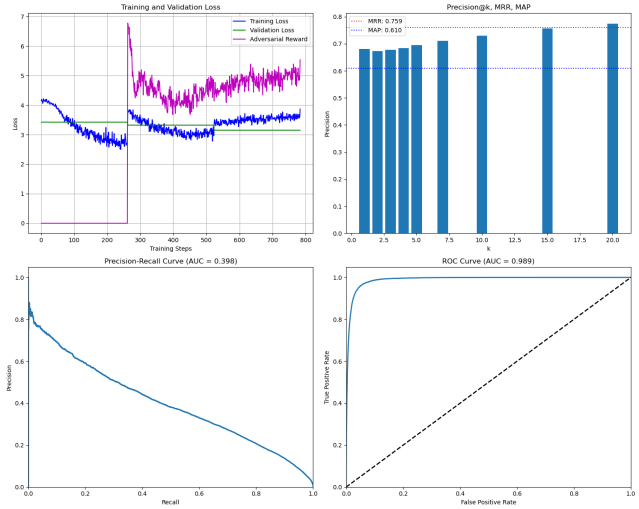


Figure 3: Performance metrics on training set

As shown in the top-left plot in 3, the training loss decreases steadily during the first epoch, indicating the model is learning meaningful representations. During the second and third epochs, as hard negatives are introduced, the training loss stabilizes and slightly increases, which is expected as the task becomes more challenging. The adversarial reward curve reflects the adversary's growing ability to identify informative hard negatives.

On ranking metrics, the model performs strongly. We achieve an MRR of 0.759 and MAP of 0.610, indicating good ranking quality and relevance of the top predictions. Precision@k improves consistently with higher values of k, reaching over 75% at k=20. The ROC curve confirms this performance with an AUC of 0.989, showing excellent separation between positive and negative pairs. The Precision-Recall curve yields an AUC of 0.398, which, although lower than ROC-AUC, remains acceptable in this sparse, highly imbalanced setting, and shows that our model is not overfitting.

4.2 Test set

Evaluation on the held-out test set (Figure 4) confirms that the model generalizes well, although there is a notable drop in performance compared to the training set.

The test set results show an MRR of 0.379 and MAP of 0.290, which, although lower than training metrics, are still significantly above random, suggesting that the model captures generalizable patterns. Precision@k improves with increasing k, reaching 57% at k=20.

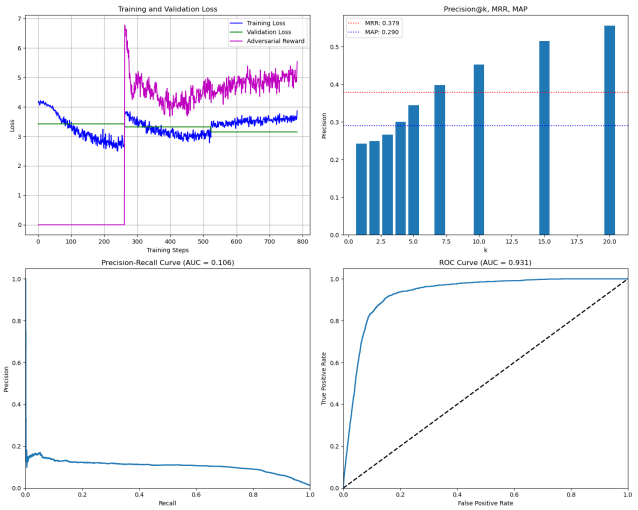


Figure 4: Performance metrics on test set

The ROC-AUC remains relatively high at 0.931, reflecting solid discrimination ability. However, the Precision-Recall AUC drops significantly to 0.106, highlighting the difficulty of maintaining high precision in this extremely sparse and imbalanced setting. This suggests that while the model can correctly distinguish pairs in aggregate (ROC), precision at lower recall levels remains a challenge.

5 Discussion

The results highlight both the strengths and limitations of our current approach.

On the training set, our model demonstrates strong learning dynamics, achieving high ranking performance (MRR = 0.759, MAP = 0.610) and excellent class separation (ROC AUC = 0.989). The introduction of hard negatives during the second and third epochs effectively increases training difficulty and improves robustness, as reflected in the adversarial reward curve and stable loss trajectory. This confirms that our adversarial sampling strategy successfully challenges the model and encourages generalization.

On the test set, while the performance drops across all metrics—as expected due to the model encountering previously unseen structures—it remains significantly above chance. The MRR of 0.379 and MAP of 0.290 confirm that the model retains reasonable ranking ability even under distribution shift. The relatively high ROC AUC (0.931) suggests that the model can still distinguish relevant from irrelevant pairs, but the sharp drop in Precision-Recall AUC (0.106) reveals the challenge of retrieving positive links in a sparse, highly imbalanced graph.

A key observation is that despite maintaining strong ranking quality (MRR, MAP), the model struggles with recall in the test set, possibly due to sparsity and the high degree variance between nodes. This may be further exacerbated by disconnected components and field-specific clustering in the graph, which increases the difficulty of generalization when entire topical clusters are excluded from training.

Future work could explore domain adaptation techniques, improved regularization, or curriculum learning strategies to better handle such graph imbalance. Additionally, leveraging structural node features or performing link prediction jointly with node classification might enhance recall without sacrificing ranking quality.

6 Conclusion

In this project, we tackled the challenging task of predicting associations between products and patents using a graph-based learning framework. Our work spanned the design of a custom data splitting strategy to account for structural imbalance and sparsity, the development of an adversarial training pipeline to improve link prediction, and a thorough evaluation of model performance.

Our results show that the model is able to learn meaningful associations on the training data and generalize to unseen test structures with moderate success. Metrics such as MRR, MAP, and ROC AUC reflect strong ranking capabilities, while the lower precision-recall performance on the test set highlights the inherent difficulty of the task due to graph sparsity and structural diversity.

Despite these limitations, our pipeline demonstrates that it is possible to build a predictive model for patent-product association with no explicit domain knowledge, using only graph structure and textual representations. Future work can explore more advanced negative sampling strategies, integrate richer node features, or incorporate external knowledge to further boost performance and generalization.

This work lays the foundation for more robust product-patent alignment systems and offers valuable insights into graph learning under real-world constraints such as sparsity, imbalance, and distributional shifts.

References

[1] Mainak Ghosh, Sebastian Erhardt, Michael E. Rose, Erik Buunk, and Dietmar Harhoff. 2024. PaECTER: Patent-level Representation Learning using Citation-informed Transformers. arXiv:2402.19411 [cs.IR] https://arxiv.org/abs/2402.19411

[2] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. *arXiv preprint arXiv:2004.11362* (2020).

[3] Abhay Kumar, Louis Owen, Nilabhra Roy Chowdhury, and Fabian Güra. 2025. ZClip: Adaptive Spike Mitigation for LLM Pre-Training. arXiv:2504.02507 [cs.LG] https://arxiv.org/abs/2504.02507

[4] Jieh-Sheng Lee and Jieh Hsiang. 2019. PatentBERT: Patent Classification with Fine-Tuning a pre-trained BERT Model. arXiv:1906.02124 [cs.CL] https://arxiv.org/abs/1906.02124

[5] Matthias Lehmann. 2024. The Definitive Guide to Policy Gradients in Deep Reinforcement Learning: Theory, Algorithms and Implementations. arXiv:2401.13662 [cs.LG] https://arxiv.org/abs/2401.13662

[6] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. 2025. Gemma 3 Technical Report. arXiv:2503.19786 [cs.CL] https://arxiv.org/abs/2503.19786