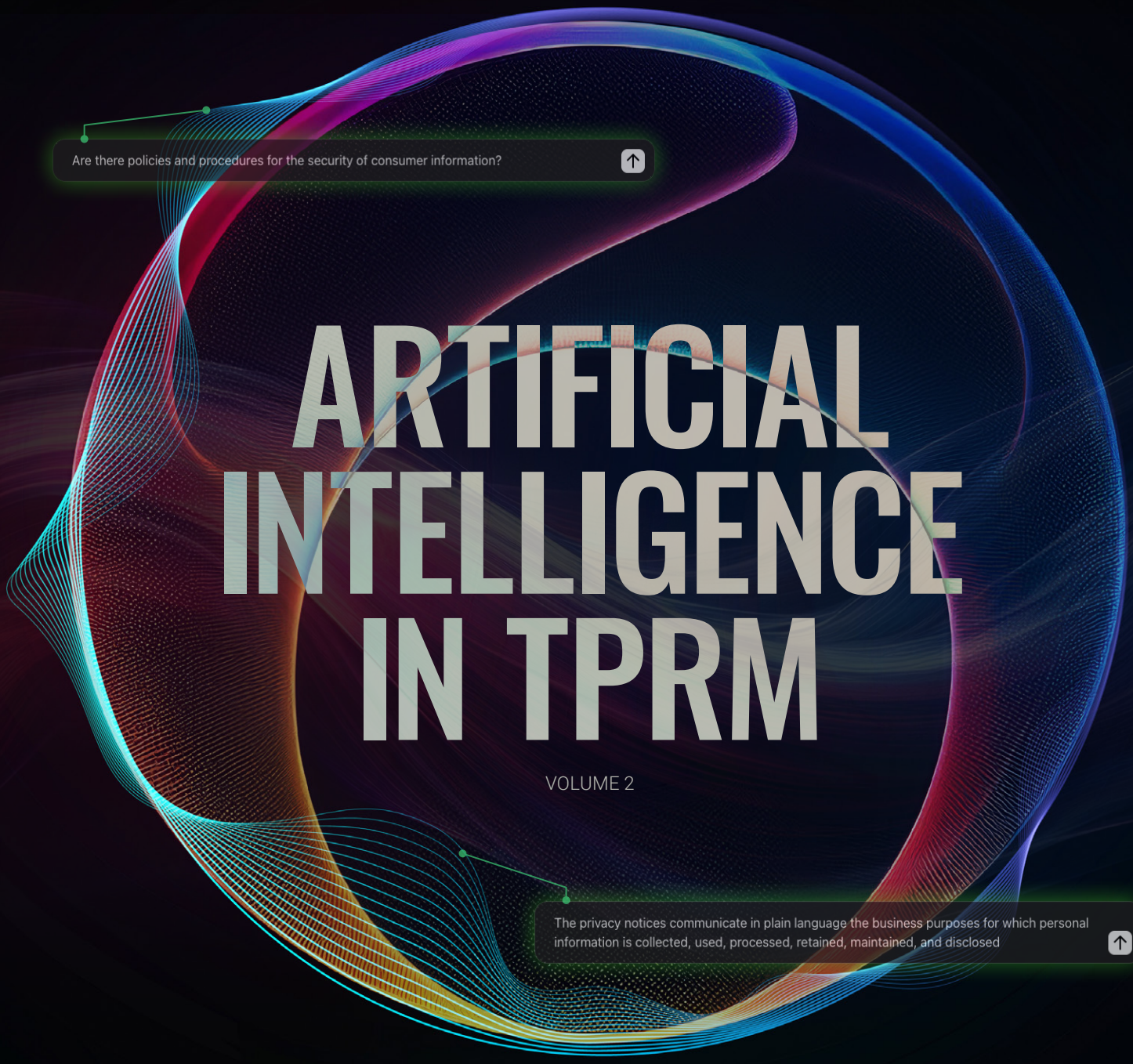
BLACK KITE



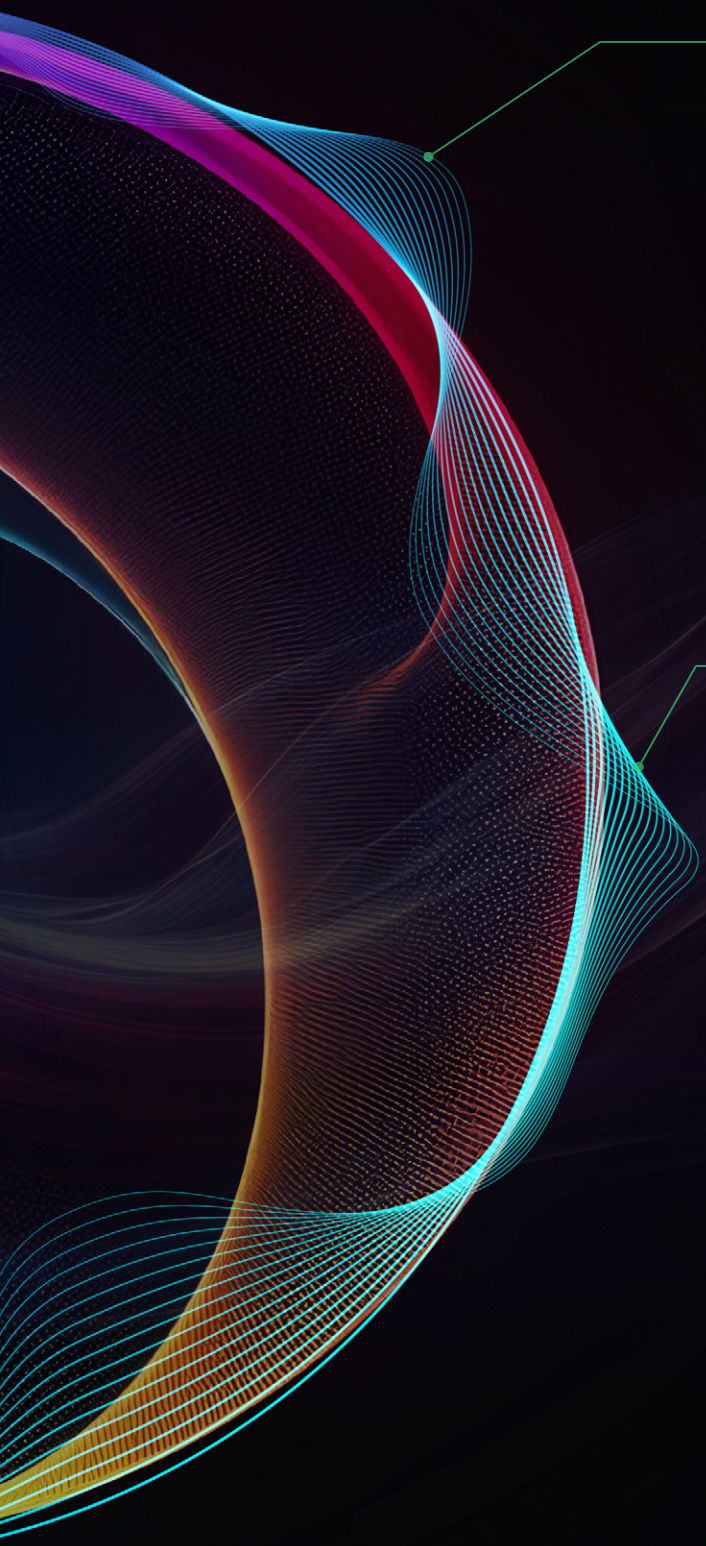
ARTIFICIAL INTELLIGENCE IN TPRM

VOLUME 2

The NLP Engineer's Guide to Building a Domain-Aware AI

Arranged by the Black Kite Research & Intelligence TEam (BRITE) • Led by Gokcen Tapkan

Table of contents



Topics Covered

- The need for a domain-specific, particularly TPRM-specific, AI language model
- The basics of LLM, transformer models and embeddings
- The recipe to creating your in-house LLM, all the way from pre-trained models to fine-tuning and testing
- The benchmarks of Black Kite's recently developed UniQuE™ Parser (v3) with respect to its predecessor (v2), ADA (OpenAI) and Gecko (Bard)

Key Findings

- Texts from cybersecurity and TPRM domain were leveraged to form pairs
- Over 20,000 pairs were labeled by Black Kite TPRM experts according to their similarity level
- Nearly 20 candidate cyber TPRM aware models developed
- The candidate base embedding models were selected from Hugging Face leaderboard
- UniQuE™ Parser 3.0 performed 10% better than OpenAI's embedding model ADA and 22% better than Google's Gecko embedding model with respect to F1 value (a combination of true positive and false positive rate)
- It also beat ADA and Gecko embedding models in terms of Spearman's rank correlation, which applies to our TPRM-specific similarity task

Recap of Volume 1

Volume 1 underscored the critical importance of integrating automation, particularly Artificial Intelligence (AI), into Third-Party Risk Management (TPRM), emphasizing the following key points, the need, the use and the challenges:

The need centered around the substantial time currently spent on manual TPRM activities. This involves collecting data from diverse platforms and knowledge bases, and curating data.

The use is obvious with the transformative potential of advanced AI technologies to redefine the landscape of third-party risk management.

The challenges in data, technology and business need to be addressed before investing in developing an in-house AI model in TPRM. The capability of AI is to proactively identify and mitigate risks inherent in third-party affiliations.

While the previous version laid the groundwork for the necessity of custom AI in TPRM, this iteration delves deeper into domain-specific AI details, specifically tailored for the TPRM sector. Here, you'll discover the essential steps for implementing task-specific or domain-specific models.

THE NEED FOR A TASK-SPECIFIC AI MODEL IN THE TPRM SPACE

Imagine an individual with a general everyday knowledge and basic English comprehension. They can grasp information from sources like Wikipedia and are familiar with basic concepts. When confronted with a statement like "Abraham Lincoln is a Hollywood actor living in LA," they can promptly identify its inaccuracy and respond with "You are wrong!" This person demonstrates an intelligence that enables them to detect inconsistencies and errors in the information they encounter.

The AI's cybersecurity knowledge is moderate, mirroring the average understanding across various domains. When terms like "physical security" and "cloud access control" are mentioned, they might respond, "You're discussing security, correct? Aren't those two concepts interchangeable?" But are they really?

While the two sentences on the right may appear as separate concepts to the average person, they are closely related within the information security domain, sharing similar underlying concepts. Additionally, in the Third-Party Risk Management (TPRM) workspace, questionnaires typically comprise either statements or actual questions, with the grammatical structure often being inconsequential. These considerations should be factored into the selection of the base Large Language Model (LLM).

A good example directly from the Information Security and TPRM domain would be the similarity regarding the following two controls:

Does Symmetric encryption use AES with a key length of at least 256 bits?

Generation of strong cryptographic keys.

An Introduction Into Large Language Models (LLMs) and Building Blocks

The surge in popularity of Generative AI (GenAI) and Large Language Models (LLMs) has dominated recent headlines. With the surge of popularity, there has been some confusion about what each term really means: these terms need clarification and revisit.

A Large Language Model (LLM) is a type of machine learning model that can perform a variety of Natural Language Processing (NLP) tasks such as generating and classifying text, answering questions in a conversational manner, and translating text from one language to another. The label "large" refers to the number of values (parameters) the language model can change autonomously as it learns.

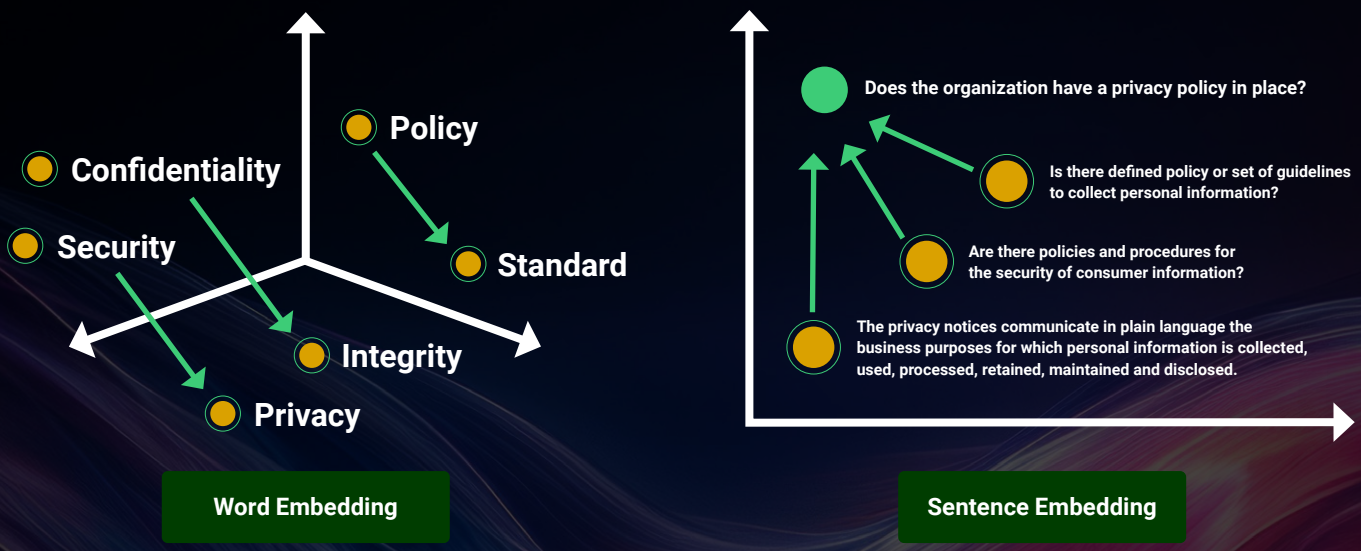
GenAI means artificial intelligence capable of generating text, images or other data using generative models often in response to prompts. Generative AI models learn the patterns and structure of their input training data and then generate new data that has similar characteristics. GenAIs are not only capable of generating text, but also image, sound, etc.

While applications like ChatGPT and Gemini have gained attention, it's essential to recognize that these tools merely begin to explore the vast capabilities of this technology. To grasp the full potential, diving into the concept of embeddings — the language of GenAI and LLMs — is crucial.

What is Embedding?

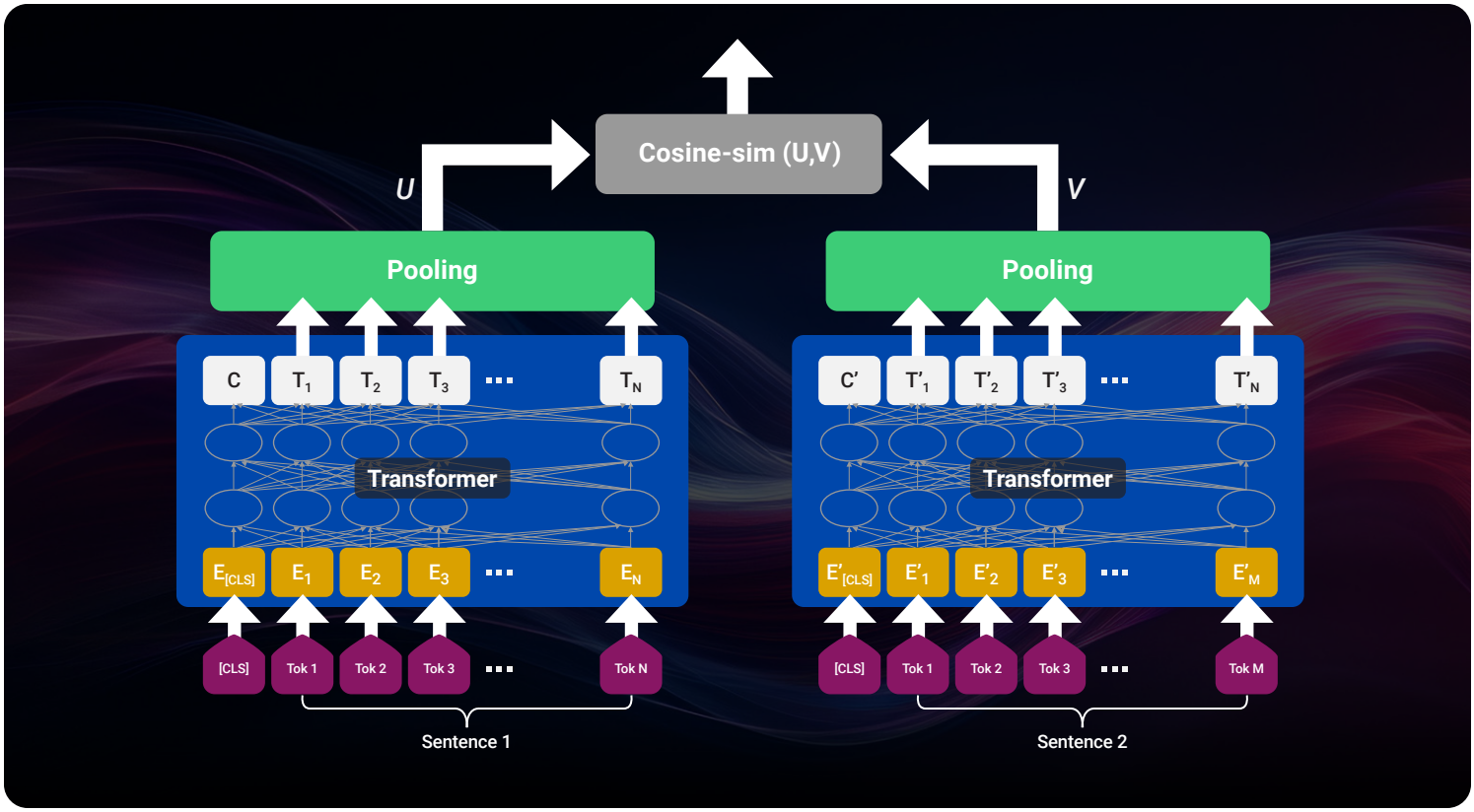
Embeddings serve as numerical representations for words, sentences, paragraphs, and even entire documents. They can be thought of as vectors in n-dimensional space.

They can be word embeddings as in V2VEC models as well as sentence embeddings.



Embeddings in Action: LLMs and GenAI

Many embedding models utilize transformers, an architecture first introduced in "Attention is All You Need" by Vaswani et al. (https://arxiv.org/abs/1706.03762). These transformers are fundamental to modern AI, focusing on "attention" to pinpoint important input for predictions. This clever mechanism enables them to effectively handle long sequences by considering the entire context.



**Siamese Sentence Transformer (STransformer) Architecture (source: https://www.researchgate.net/figure/Siamese-Sentence-Transformer-STransformer-Architecture_fig2_353487642)*

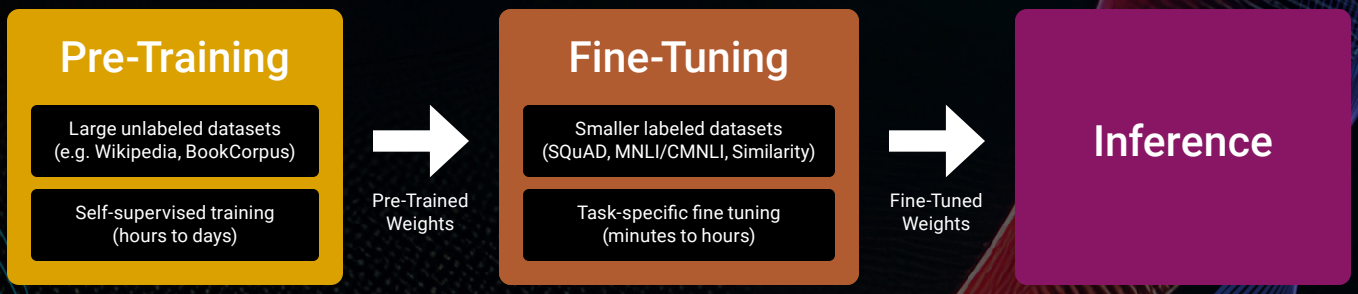
Large Language Models (LLMs) are AI models tailored for tasks involving language understanding. They employ transformer models to convert input data into embeddings, which are then used in predictive models like recurrent neural networks (RNNs) or long short-term memory (LSTMs). By understanding the context and relationships between words in the vectors, the model can generate the most likely output based on its training data.

While LLMs are specialized for language tasks, similar Generative AI (GenAI) models can be applied to diverse tasks like turning text into images or converting audio to text. These models, regardless of their specific use, interpret the meaning of their input and, using embeddings, produce the most probable output based on their training.

Large Language Models (LLMs) are AI models tailored for tasks involving language understanding.

Transforming Your Data Into Embeddings: Fine-Tuning

In recent times, models such as BERT, GPT, Mistral, and LLAMA have brought Large Language Models (LLMs) into the spotlight. While various models can generate embeddings, one straightforward method we'll explore is fine-tuning a pre-trained model to transform your data into embeddings.



**Pre Training and Fine Tuning example. (Source:https://docs.graphcore.ai/projects/bert-training/en/latest/bert.html)*

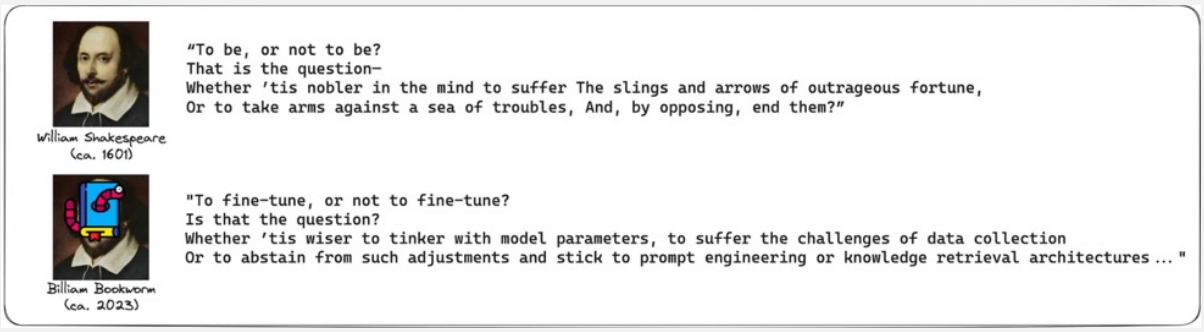
Pre-trained models offer a quick and efficient way to harness the power of embeddings, which are numerical representations of words or phrases that capture their meanings. With models like Generative Pre-trained Transformers (GPT), you can easily convert text data into embeddings. Once you have these embeddings, you can use them for a variety of tasks, such as finding similar items in a dataset or improving recommendation systems.

While the initial training of Large Language Models (LLMs) involves millions or even billions of iterations, the process of tailoring these models for a particular task or domain demands a more targeted approach. This is where the concept of fine-tuning becomes crucial.

Fine-tuning takes the process a step further. It involves customizing a pre-trained model to better suit your specific needs. One way to do this is by adding more examples to the training data, allowing the model to learn from a broader range of contexts. For instance, if you have company-specific terms or jargon, fine-tuning a model with more examples from your company's domain can make it better at understanding and handling language specific to your business.

Additionally, fine-tuning can be task-specific. For example, consider a scenario where you want a model to assist with coding tasks. By providing it with context-specific examples like code snippets or comments, you can fine-tune the model to become more adept at tasks such as code completion or suggesting improvements.

In essence, pre-trained models provide a convenient starting point, and fine-tuning allows you to tailor these models to better fit your unique requirements, whether that involves understanding industry-specific language or excelling in particular tasks like coding assistance.



** Times they are a changin (Source: https://blog.ml6.eu/to-fine-tune-or-not-to-fine-tune-837996713913)*

Fine-tuning is a crucial step in the process of utilizing pre-trained models because it enables customization and adaptation to specific contexts or tasks. While pre-trained models like GPT offer a solid foundation by learning from a vast amount of general data, they may not be optimized for your particular use case out of the box.

Here's why fine-tuning is necessary:

Domain-Specific Knowledge: Pre-trained models may lack familiarity with domain-specific terms, jargon, or nuances relevant to your industry or organization. By fine-tuning with examples from your domain, the model gains a more nuanced understanding of your specific language.

Task-Specific Adaptation: Pre-trained models are taught a diverse set of tasks, but they might not excel at your specific task without fine-tuning. For instance, if you're working with a recommendation system, fine-tuning allows the model to understand dataset-specific patterns, leading to more accurate and relevant recommendations.

Enhanced Performance: Fine-tuning with additional examples enhances the model's performance by exposing it to a more extensive range of scenarios. This helps the model generalize better and make more informed predictions in your unique context.

Improved Efficiency: Fine-tuned models are often more efficient in terms of computational resources. They require less training time compared to training a model from scratch, making them a practical choice for many real-world applications.

Task-Specific Fine-Tuning: In certain cases, you might want to specialize a pre-trained model for a specific task within a broader domain. For example, if you're using the model for code completion, fine-tuning with task-specific examples ensures the model understands the intricacies of coding language and syntax.

In summary, fine-tuning is essential for tailoring pre-trained models to your specific needs, making them more proficient in understanding your language, adapting to your tasks, and ultimately improving their overall performance in your application.

Data Engineering: It All Starts With Data

The training of LLMs is a fascinating process that commences with an extensive and diverse dataset. This dataset can be derived from various sources such as customers, user feedbacks, domain experts (if saved).

The key emphasis lies in forward-thinking data management, capturing user behaviors and expert insights. Such foresight proves to be a valuable investment for future endeavors.

Invest in Data, Secure Your Future

After assembling the dataset, it's not immediately ready for use—similar to a raw diamond, it requires polishing. This entails the crucial steps of cleaning and preparing the data. This process may involve tasks such as converting the dataset to lowercase, eliminating stop words, and breaking down the text into smaller components known as tokens. These tokens essentially serve as the fundamental building blocks of language.

[This] [is] [token] [izing] [.]

The NLP Engineer's Guide to Build a Custom LLM

ASSEMBLING THE DATASET

- The dataset, akin to a raw diamond, requires polishing before use.

DATA CLEANING AND PREPARATION

- Cleaning and preparing the data.
- If needed, convert the dataset to lowercase.
- Try different grammatical structures, such as questions, neutral statements, negative statements.
- Try repetition, so that your LLM does not forget what it has learned in the beginning.

TOKENIZATION

- Understand the tokenization of your base model.
- Enlarge tokens if needed, add new tokens.

DECIDE ON YOUR BASE MODEL

- Leverage Hugging Face, forums, benchmarks.
- Decide on a task-specific LLM or a generic LLM.

FINE-TUNE

- Read the fine-tuning best practices.
- Decide on your batch numbers, loss function, evaluation function and optimizer.

TEST, TEST AND TEST

- Test in different datasets.
- Do not just rely on numbers like F1, TPR, FPR, Spearmanr, but hire human experts so that your LLM is assessed on individual use cases.

Labeling

In the realm of model training, the labeling process serves as the foundation for the machine learning algorithm's understanding of patterns and contexts within your domain. The substantial size of the labeled dataset, ranging from 20,000 to 30,000 pairs, is not arbitrary; it is a deliberate effort to expose the model to a diverse range of examples, ensuring it captures the intricacies of your specific domain.

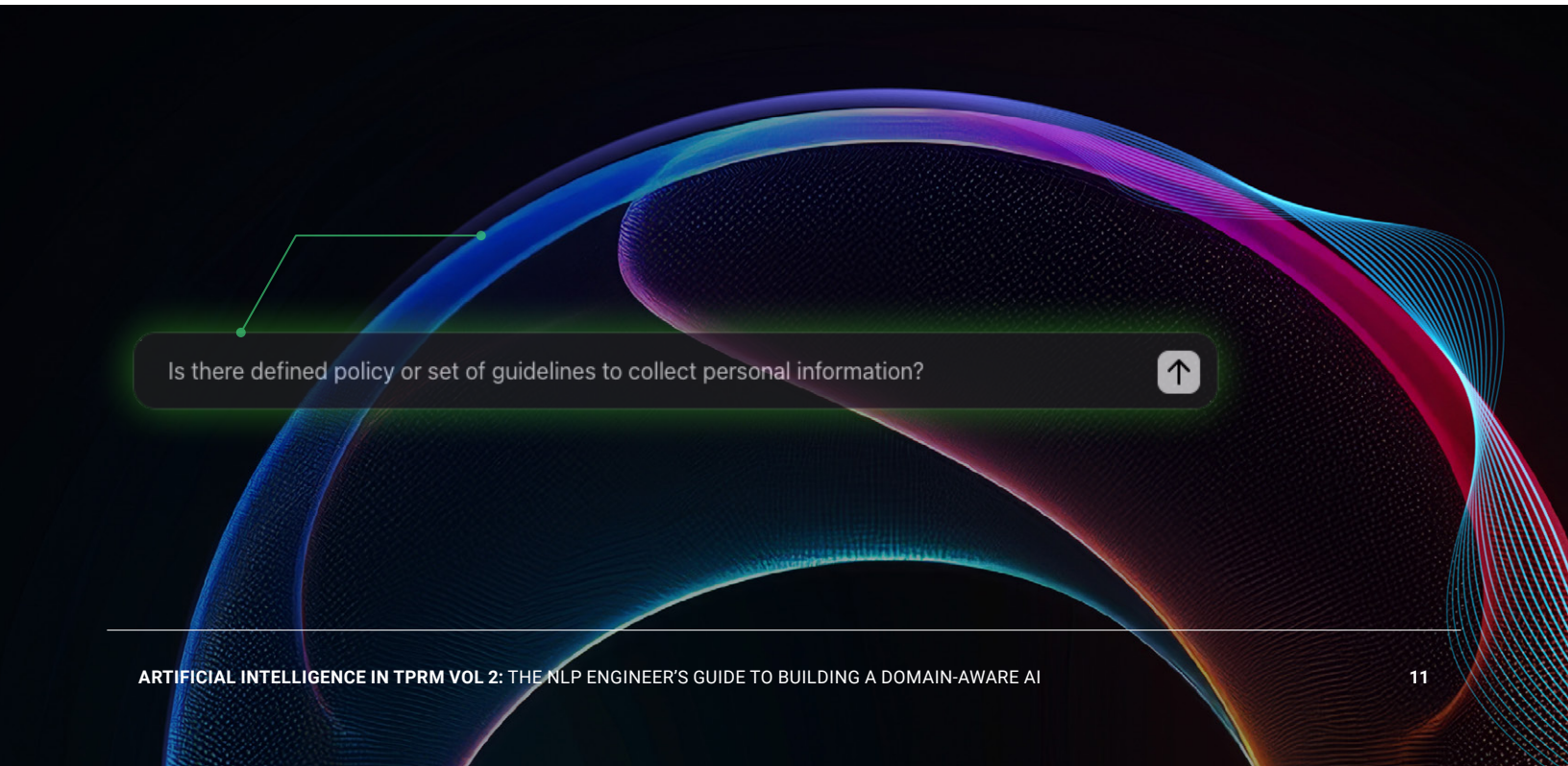
Each labeled pair contributes to the model's learning process, and any inaccuracies or oversights in labeling could potentially compromise the effectiveness of the entire training phase.

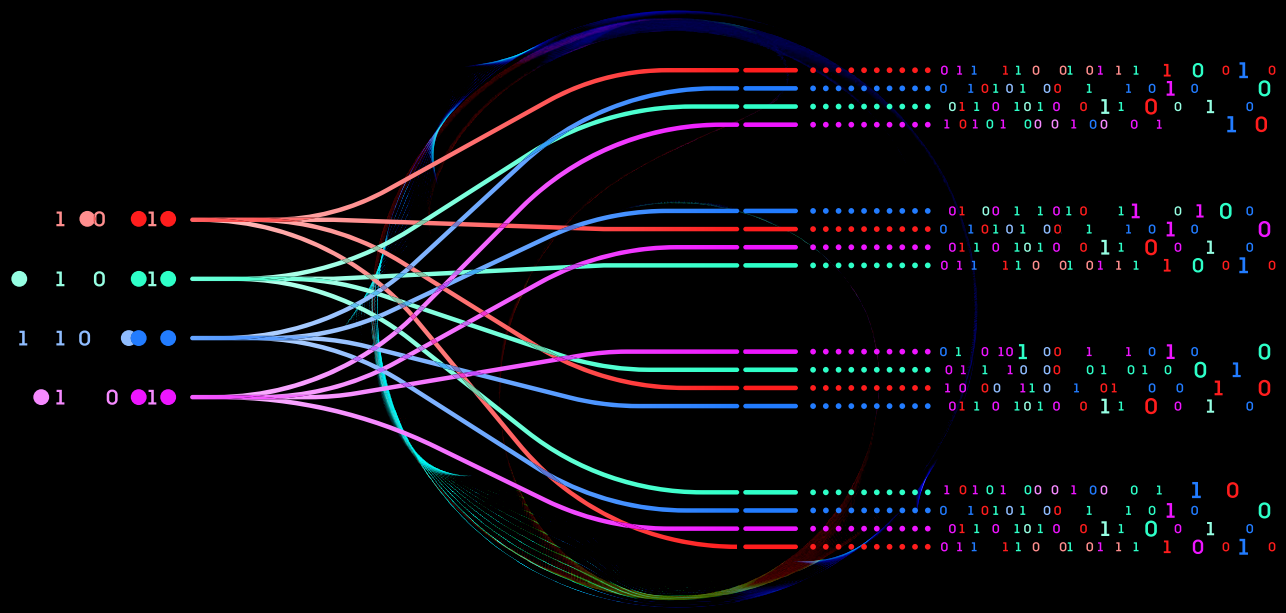
Understanding the reasons behind the scale of data labeling and the challenges it poses sheds light on the commitment required to achieve robust model performance. Despite the costs and time investments, the precision attained through manual labeling is often unmatched, providing a solid foundation for the model's ability to generalize and make accurate predictions in real-world scenarios.

Navigating the complexities of the labeling process may prompt considerations for strategies to streamline the effort. Exploring tools or methodologies that can optimize the labeling workflow could prove invaluable, potentially mitigating costs and time constraints while maintaining the necessary level of precision.

In summary, the scale and precision of the data labeling process are pivotal elements in shaping the success of your machine learning model. This nuanced understanding not only emphasizes the importance of the endeavor but also provides insights for practitioners seeking practical approaches to tackle the challenges inherent in preparing labeled datasets for effective model training.

The manual labeling of data can be a resource-intensive task.





Landscape of Large Language Models (LLMs)

State-of-the-Art Large Language Models: SBERT – Sentence-BERT

The transformative impact of transformers on Natural Language Processing (NLP) is undeniable. This progress has paved the way for the development of various machine learning models, with BERT standing out as a prominent example. Built upon the transformer architecture, **BERT** comprises stacked transformer encoders and has found applications in diverse problem domains such as sentiment analysis and question answering.

The impact of transformers on Natural Language Processing is undeniable.

One notable contribution of BERT is its role in constructing word embeddings—numeric vectors that represent the semantic meanings of words. This shift to representing words as embeddings is advantageous, as it enables machine learning algorithms to work with vectors rather than raw text. This, in turn, facilitates the comparison of words based on similarity, utilizing metrics like Euclidean or cosine distance.

While BERT excels at word-level embeddings, practical applications often require embeddings for entire sentences. The challenge arises because the basic BERT version focuses solely on the word level. To address this limitation, several BERT-like approaches have been developed, aiming to generate embeddings for entire sentences. In this article, we will explore these solutions, culminating in a discussion about the state-of-the-art model known as SBERT.

Open-Source AI Large Language Models (LLMs)

In today's digital age, language, once confined to the realms of human cognition and communication, is now gradually being conquered by machines. Leading this transformative era are Large Language Models (LLMs). These powerhouse computational models, honed through extensive textual data, have become highly proficient in comprehending, generating, and interacting using human language across a spectrum of applications. From simplifying tasks such as answering queries and producing text to more nuanced roles like sentiment analysis, content recommendation, and even creative writing, LLMs are revolutionizing the scope of artificial intelligence.

The significance of LLMs transcends their linguistic prowess. They serve as proof to the progress in machine learning, data processing, and cloud computing. As these models evolve in complexity and scale, they hold tremendous potential across various sectors, including healthcare, education, entertainment, and business. Furthermore, with the ethos of open source becoming a cornerstone in AI development, the democratization of LLMs is poised to foster unparalleled levels of innovation, collaboration, and accessibility.

In the vast landscape of open-source AI models, Large Language Models (LLMs) have emerged as some of the most sought-after tools, capable of generating natural language texts across diverse tasks and domains. Here's a glimpse of some of the top open-source libraries and frameworks that deserve your attention:

1. **Hugging Face Transformers:** Hugging Face provides a repository of pre-trained models for Natural Language Processing. The Transformers library simplifies the integration of these models into your projects, offering a convenient resource for developers.
2. **BERT (Bidirectional Encoder Representations from Transformers):** Developed by Google, BERT is an influential open-source pre-trained language representation model. It has made a significant impact on various natural language understanding tasks, contributing to the advancement of language processing capabilities.
3. **TensorFlow:** An open-source software framework developed by Google, TensorFlow is widely used for building and deploying machine learning neural networks. With a user-friendly setup and extensibility, TensorFlow boasts a large and active community of developers and researchers. It supports diverse model types, including LLMs, computer vision, and reinforcement learning.
4. **PyTorch:** Developed by Facebook, PyTorch is an open-source machine learning library based on the Torch library. Renowned for its dynamic computation graphs, distributed training capabilities, and a rich set of tools and libraries, PyTorch is a preferred choice for developing deep learning models, especially LLMs.
5. **Scikit-Learn:** This open-source machine learning library for Python, built on top of NumPy, SciPy, and Matplotlib, provides a simple and consistent interface for various machine learning tasks. Through its integration with other libraries like TensorFlow and PyTorch, Scikit-Learn supports a variety of models, including LLMs.

Whether you're deep-diving into Natural Language Processing or broader machine learning applications, these tools offer a wealth of possibilities.

Results and Experiments on UniQuE™ Parser 3.0 and Well-Known Embedding Models

What is UniQuE™ Parser 3.0?

We initially developed an embedding model known as UniQuE™ Parser 2.0, a one-of-a kind tool to parse compliance documents. In the spirit of continual improvement we recently upgraded the Parser to a new version: Parser 3.0. This latest version is not just an update; it's significantly more powerful and efficient than Parser 2.0.

The need for such an upgrade was driven by the ever-evolving landscape of data processing and the increasing complexities in the cybersecurity domain. UniQuE™ Parser 3.0 is specifically designed with these advancements in mind. It is equipped to handle and understand the nuances of cybersecurity and Third-Party Risk Management (TPRM) areas much better.

Our commitment to maintaining the most up-to-date tools led us to develop UniQuE™ Parser 3.0, ensuring that our embedding model remains at the forefront of technology, capable of delivering more accurate and relevant results in the field of cybersecurity and beyond.

Benchmark on UniQuE™ Parser 3.0 with Well-Known Embedding Models

In the evolving landscape of Natural Language Processing (NLP), the search for more efficient and accurate models remains paramount. Our team selected the most promising models for comparison: the ADA and Gecko embedding models. These models stand out in text processing, each backed by giants in the field of artificial intelligence—OpenAI and Google, respectively.

To enhance the robustness of our evaluation, we integrated a cold dataset representing the Third-Party Risk Management (TPRM) domain. This dataset, specifically curated to embody the concepts associated with TPRM, provided a perspective to assess the models' capabilities in handling specialized, domain-specific content.

ADA: A Creation of OpenAI

ADA, developed by OpenAI, is renowned for its robustness and adaptability in text processing tasks. It leverages advanced machine learning techniques to understand and generate human-like text, making it a cornerstone for projects that require deep semantic understanding.

Gecko: Google's Answer to Text Embedding

On the other hand, Gecko is Google's contribution to the NLP community, designed to excel in embedding textual information into high-dimensional spaces. This allows for nuanced detection of patterns and relationships within text, facilitating more sophisticated processing tasks.

Both ADA and Gecko are at the forefront of embedding models, offering powerful tools for a myriad of text processing applications. Their widespread use underscores their effectiveness and the trust they have gained within the tech community.

To ascertain which model best suits our needs, we conducted a benchmarking exercise, integrating both ADA and Gecko with our Parser 3.0 candidate models. This comparison was structured to evaluate a comprehensive set of performance metrics, including:

- **Mean Square Error (MSE):** To measure the average of the squares of the errors between predicted and actual values.
- **Precision:** To assess the accuracy of the models in identifying relevant data points.
- **True Positive Rate (TPR):** To determine the proportion of actual positives correctly identified.
- **False Positive Rate (FPR):** To calculate the ratio of negative instances incorrectly classified as positive.
- **F1 Score:** To evaluate the balance between precision and recall.
- **Spearman's Rank Correlation Coefficient (Spearmanr):** To measure the strength and direction of the association between the ranked variables.

This rigorous analysis aims not only to highlight the strengths and weaknesses of each model but also to guide our selection of the most suitable embedding model for enhancing our Parser 3.0's capabilities. By doing so, we ensure that our technology remains at the cutting edge, capable of delivering unparalleled accuracy and efficiency in text processing tasks.

The below graph is the result of scores from each model.

Model Name	Mean Square Error	Precision	TPR	FPR	F1	Spearmanr
UniQuE™ Parser 3.0 Candidate1	0.17	0.71	0.62	0.14	0.66	0.58
UniQuE™ Parser 3.0 Candidate2	0.19	0.71	0.61	0.14	0.66	0.53
ADA Embedding Model	0.42	0.68	0.54	0.14	0.60	0.50
Gecko Embedding Model	0.32	0.64	0.46	0.14	0.54	0.45

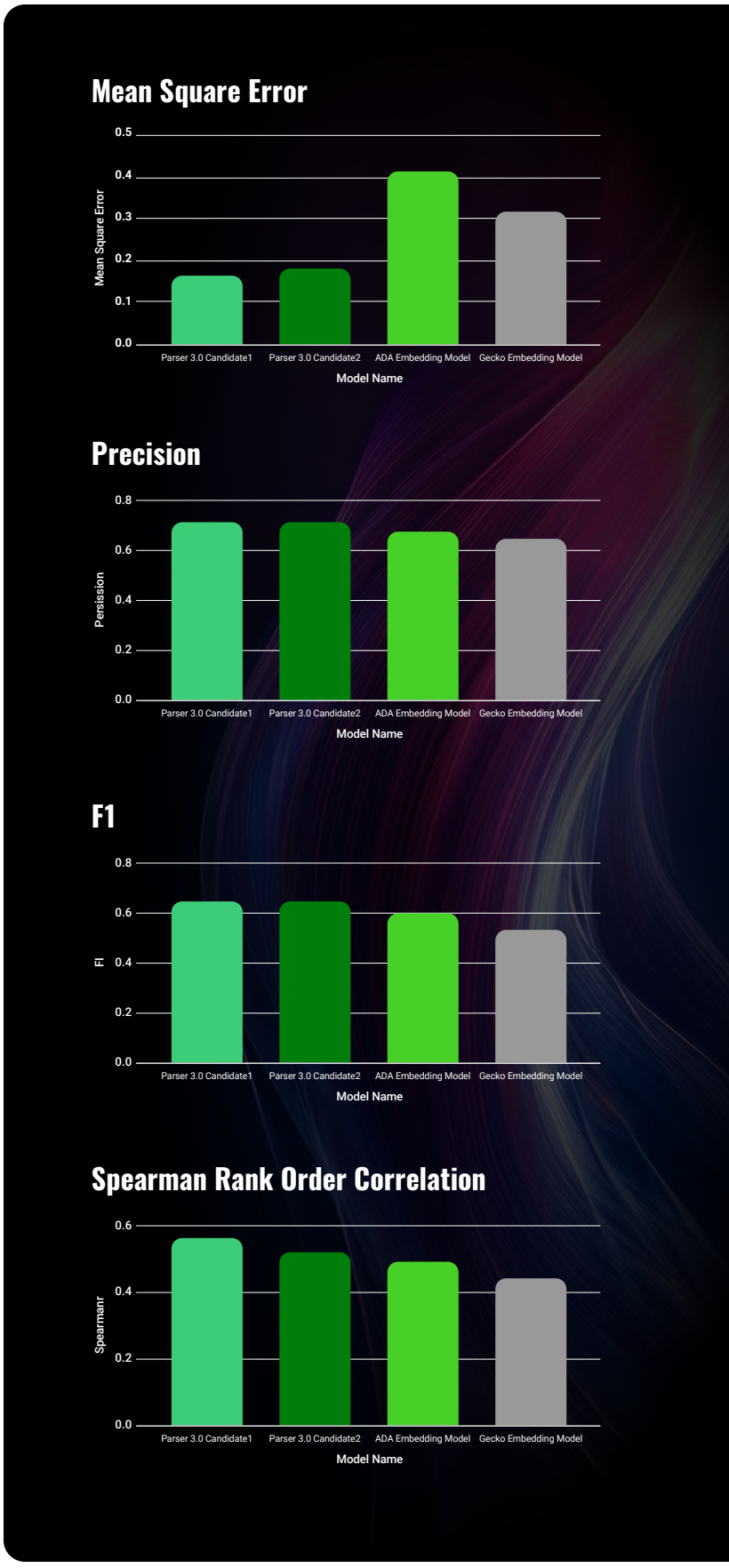
In our above benchmarking exercise, we compared two leading embedding models, ADA from OpenAI and Gecko from Google, against our proprietary candidates for the Parser 3.0 tool. The evaluation focused on a spectrum of performance metrics crucial for text processing effectiveness.

The results revealed that Parser 3.0 Candidate 1 demonstrated superior performance across all metrics. It boasted the lowest Mean Square Error (0.167), indicating its predictions were closest to the actual values. This candidate also achieved the highest precision (0.7103), suggesting a superior rate of relevant instance prediction, and it led in True Positive Rate (0.6179), pointing to its effectiveness in identifying true positives. In the below there are detailed graph for the mean square error and precision scores.

Consistency was observed in the False Positive Rate (0.14) across all models, suggesting a similar propensity for false alarms. However, Parser 3.0 Candidate 1 maintained its lead with the highest F1 Score (0.66), which balances precision and recall, and the strongest Spearman's Rank Correlation Coefficient (0.57), indicating the most accurate ranking capability. At right are detailed graphics for the F1 and Spearmanr scores.

In comparison, the ADA model showed the highest Mean Square Error, while Gecko outperformed ADA in this aspect but still lagged behind both Parser 3.0 candidates in the other metrics. These insights underscore Parser 3.0 Candidate 1 as the most promising model for our text processing applications, given its comprehensive outperformance in our rigorous evaluation.

According to our benchmark results, we can say that our candidate models outperform the ADA and Gecko embedding models.



Conclusion

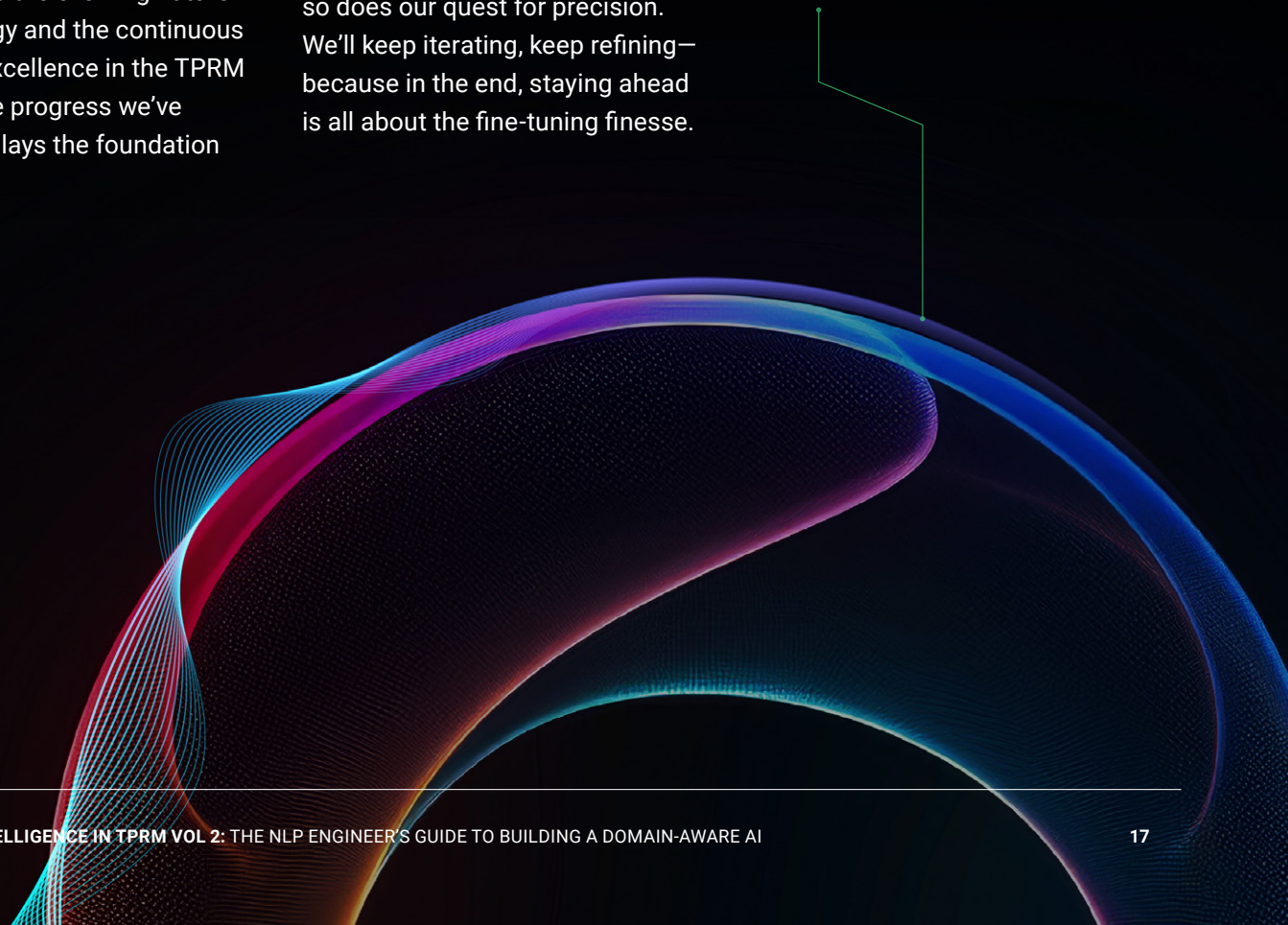
In summary, our exploration into the realm of AI, particularly within the Third-Party Risk Management (TPRM) space, affirms the indispensable nature of task-specific models.

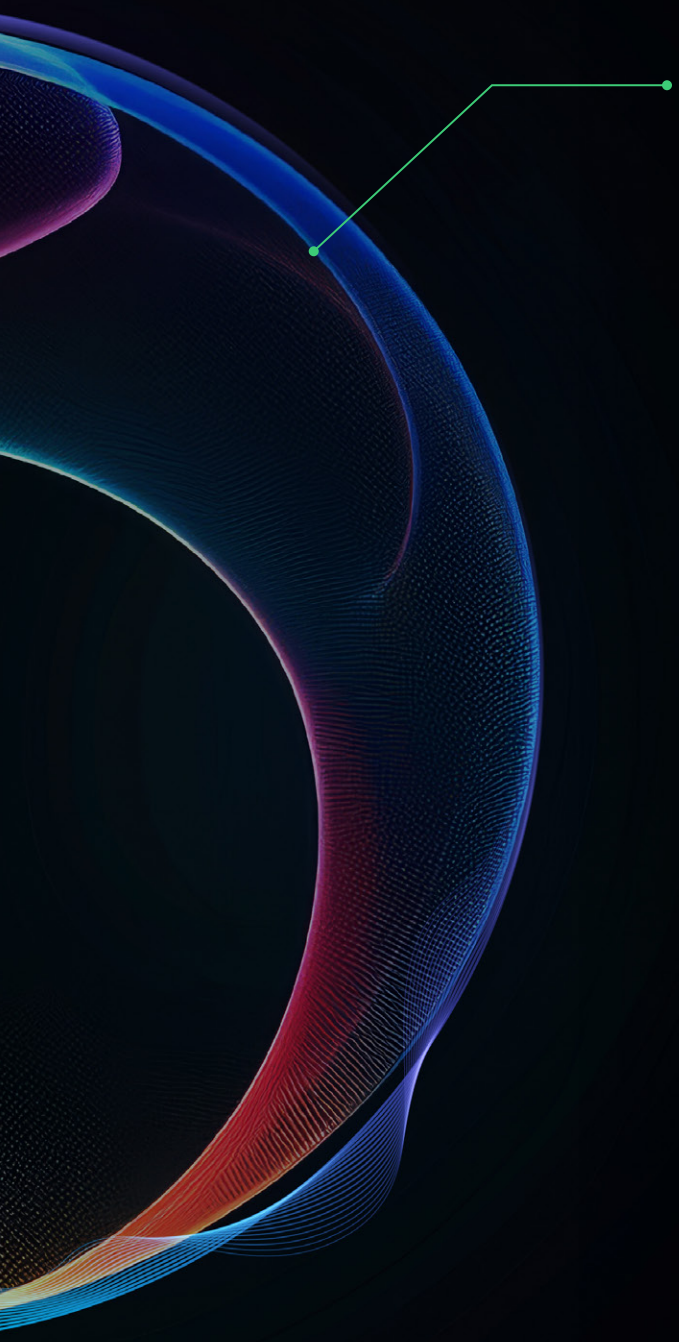
The journey of embedding models from concept to implementation is marked by a continuous cycle of refinement. Our benchmarks demonstrate that while ADA and Gecko provide strong foundations, our Parser 3.0 candidates, fine-tuned for our unique requirements, lead the way in performance.

We are reminded through this process that the art of fine-tuning is as much about the journey as it is the destination. It's a testament to the evolving nature of technology and the continuous pursuit of excellence in the TPRM domain. The progress we've made today lays the foundation for the innovations of tomorrow, ensuring that our models not only understand "security" in all its forms but can discern and adapt to the ever-changing landscape of cybersecurity. Fine-tuning, therefore, isn't just a technical necessity; it's a commitment to perpetual growth and improvement.

In wrapping up, it's clear that in the domain of model development, fine-tuning is not a one-off triumph but a perpetual cycle. As our benchmarking showcases, today's leaders can become tomorrow's learners. The UniQuE™ Parser 3.0 candidates currently lead the pack, yet the journey doesn't end here. Just like technology evolves, so does our quest for precision. We'll keep iterating, keep refining—because in the end, staying ahead is all about the fine-tuning finesse.

As we advance, the cyclical process of fine-tuning remains pivotal. It's akin to a gardener pruning a bonsai, making careful cuts to guide its growth. Our task-specific AI models thrive on this meticulous cultivation, becoming more precise and efficient with each iteration.





About Black Kite

Get Proactive | Gain Control.

Black Kite gives companies a comprehensive, real-time view into cyber third-party risk so they can make informed and proactive risk decisions that help avoid business disruption, building resilience within their supply chain. With one-of-a-kind collaboration capabilities, companies can work directly with their vendors to report, mitigate, and minimize risk, improving their own business resilience as well as their vendors' organizations.

Through an automated process, and a combination of threat, business, and risk information, Black Kite provides cyber risk detection and response capabilities that are accurate, fast, and transparent.

Black Kite serves more than 2,000 customers in a wide range of industries and has received numerous industry awards and **recognition from customers**.

LEARN MORE AT BLACKKITE.COM, AND ON THE BLACK KITE BLOG.

BLACK KITE

800 Boylston Street, Suite 2905, Boston, MA 02199

+1 (571) 335-0222 | info@blackkite.com

Copyright © 2024 Black Kite, Inc. All rights reserved. All other brand names, product names, or trademarks belong to their respective holders.