ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

Isotropy Matters: Soft-ZCA Whitening of Embeddings for Semantic Code Search

Andor Diera[1] and Lukas Galke[2] and Ansgar Scherp[1] *

1- Ulm University - Data Science and Big Data Analytics
Helmholtzstrasse 16, 89081 Ulm - Germany

2- University of Southern Denmark - Dept of Mathematics and Computer Science
Campusvej 55, 5230 Odense - Denmark

Abstract. Low isotropy in an embedding space impairs performance on tasks involving semantic inference. Our study investigates the impact of isotropy on semantic code search performance and explores post-processing techniques to mitigate this issue. We analyze various code language models, examine isotropy in their embedding spaces, and its influence on search effectiveness. We propose a modified ZCA whitening technique to control isotropy levels in embeddings. Our results demonstrate that Soft-ZCA whitening improves the performance of pre-trained code language models and can complement contrastive fine-tuning.

1 Introduction

Isotropy in language models (LMs) refers to the uniform distribution of vector representations in the embedding space [1]. It enhances the efficient use of the embedding space and increases robustness to perturbations. Anisotropy, i. e., when vectors are unevenly distributed, can hinder model performance on semantic tasks by making it difficult to distinguish between different meanings [2]. An anisotropic embedding space poses an even greater challenge for cross-lingual tasks, where accurate semantic alignment demands more precise representational distinctions [3]. These representational challenges extend to code LMs, notably affecting semantic code search, where natural language queries are used to retrieve relevant code snippets [4]. In this task, high anisotropy can lead to suboptimal retrieval performance, as the encoded representations of semantically different code snippets may not be adequately distinguished. The prevalence of programming language keywords and symbols intensifies this problem, as these elements can dominate the sequence representations and obscure the semantic content of the code [5].

Contrastive fine-tuning is a common approach for improving semantic code search by encouraging the model to bring semantically similar code representations closer while pushing dissimilar ones apart [6, 7]. However, fine-tuning only marginally mitigates the anisotropy problem, as it does not fully address the underlying issue of the generally low angular distance between encoded representations [8]. In NLP, multiple approaches have been proposed to improve the

*This work was co-funded by the Deutsche Forschungsgemeinschaft (DFG) as part of the CodeInspector Project - P5311038. The code for our experiments is available at https://github.com/drndr/code_isotropy.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

isotropy of an embedding space. Regularization methods [3, 2] and simple post-processing techniques [1, 9] have shown promise in enhancing the isotropy of encoded representations. ZCA whitening [10] has shown to be a particularly fitting post-processing method for decorrelating the hidden features and increasing the isotropy of embeddings [9].

As vector databases become increasingly central to modern search systems, there is growing interest in lightweight post-processing techniques that can boost performance. However, these techniques are unexplored in the context of code search tasks. To address the challenge of anisotropy in semantic code search, we analyze the embeddings space of three pre-trained code LMs: CodeBERT [11], CodeT5+ [6], and Code Llama [12]. We examine how evenly distributed (i. e., isotropic) their embeddings are, specifically looking at how this affects their performance on code search. We introduce Soft-ZCA, an extension to ZCA whitening which permits control over the degree of whitening. We evaluate our approach on six popular programming languages and test the generalization capabilities on a low-resource programming language dataset. Our analysis shows that, similarly to standard LMs, code LMs also showcase a high level of anisotropy. We confirm that contrastive fine-tuning does not have a strong effect on isotropy and demonstrate that applying Soft-ZCA whitening with an eigenvalue regularizer can improve both isotropy and code search performance. The main contributions of our paper are:

- We analyze the isotropy of the embedding spaces in three popular code LMs regarding code search performance.
- We introduce a regularizer to ZCA whitening to control the degree of isotropy in embeddings, which we call Soft-ZCA.
- Experiments on six popular and one low-resource programming language show that post-processing the embeddings with Soft-ZCA whitening improves code search for pre-trained and fine-tuned code LMs.

2 Whitening of the Embeddings

Whitening is a common processing step in machine learning and statistical analysis to transform variables or features to orthogonality [13]. Given a set of embeddings $\boldsymbol{Z} \in \mathbb{R}^{\boldsymbol{N} \times \boldsymbol{d}}$, a whitening transformation can be denoted as $\boldsymbol{H} = \boldsymbol{W}\boldsymbol{Z}^\top$, where $\boldsymbol{W} \in \mathbb{R}^{\boldsymbol{d} \times \boldsymbol{d}}$ is the square whitening matrix, and $\boldsymbol{H} \in \mathbb{R}^{\boldsymbol{N} \times \boldsymbol{d}}$ is the whitened embedding. Since the only condition of a whitening transformation is to satisfy $\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\top = \boldsymbol{I}$ (where $\boldsymbol{\Sigma}$ is the covariance matrix of $\boldsymbol{Z}$), there are infinitely many possible whitening transformations due to rotational freedom. In practice, the most widely used whitening transformations are based on Principal Component Analysis (PCA), Zero-phase Component Analysis (ZCA), or the Cholesky decomposition of the covariance matrix, each offering different properties and tradeoffs for various applications [13]. ZCA whitening [10] has been shown to maintain the highest correlation with the original data [13] and is considered the most appropriate for embedding spaces. The whitening matrix of

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

ZCA is defined as $\boldsymbol{W}^{\mathrm{ZCA}} = \boldsymbol{\Sigma}^{-1/2}$. Using singular value decomposition, $\boldsymbol{\Sigma}^{-1/2}$ can be rewritten to $\boldsymbol{\Sigma}^{-1/2} = \boldsymbol{U}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{U}^{\top}$ where $\boldsymbol{U}$ is an orthogonal matrix based on the eigenvectors of $\boldsymbol{\Sigma}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix of $\boldsymbol{\Sigma}$'s eigenvalues.

Soft-ZCA Whitening To control the degree of whitening, we introduce an eigenvalue regularizer ϵ. This adjustment modifies the whitening matrix calculation to $\boldsymbol{W}^{\mathrm{ZCA}} = \boldsymbol{U}(\boldsymbol{\Lambda}+\epsilon\boldsymbol{I})^{-1/2}\boldsymbol{U}^{\top}$, where $\boldsymbol{I}$ is the identity matrix. The key purpose of ϵ is to retain more of the original signal and variance. If any of the eigenvalues in $\boldsymbol{\Lambda}$ are close to 0, their inverse square root will become exceedingly large, which causes the whitening transformation to amplify noise and insignificant components in the data. By placing a lower bound on the eigenvalues of $\boldsymbol{\Sigma}^{-1/2}$, ϵ can directly influence the strength of the whitening transformation.

3 Experimental Apparatus

Datasets We experiment using two datasets. The first is CodeSearchNet [4], a benchmark for studying the code search capabilities of machine learning models. It encompasses code-comment pairs from six popular programming languages: Python, Go, Java, JavaScript, Ruby, and PHP. The full corpus comprises 2 million code-comment pairs. To evaluate generalization to a low-resource language, we use the StatCodeSearch test dataset [7], which comprises 1,070 code-comment pairs from social science research in the R language.

Models We investigate the embeddings of three code LMs. CodeBERT [11] is an encoder-only LM developed for programming language understanding. CodeT5+ [6] is an encoder-decoder LM trained for both code understanding and generation. Code Llama [12] is the code-specialized version of Llama 2, used predominantly for generation. In addition to the base models (pre-trained only), we include a fine-tuned CodeBERT that is trained on the code-comment pairs for each search task. Characteristics of each model can be seen in Table 1.

Table 1: Model details

Model	Number of Parameters	Embedding Dimension	Supported Progr. lang.	Contrastive Pre-training
CodeBERT	125m	768	6	no
CodeT5+	110m	256	9	yes
Code Llama	7b	4,096	7	no

Procedure We process code and natural language inputs independently, mirroring real-world search systems. Each sequence is cut off at 256 tokens with no padding added. For CodeBERT and Code LLama, we extract the sequence representations by applying mean pooling on the last hidden state. For the CodeT5+ model, we rely on the default pooling, which includes an additional

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

down-projection. We fine-tune CodeBERT for each dataset separately with InfoNCE [14] as a contrastive loss, a learning rate of 5e-5, and a batch size of 32 for 5 epochs. The whitening matrices are calculated independently for code and comments using the full test sets. We employ IsoScore [15] to measure the isotropy of the embedding space. It is bounded to $[0, 1]$, where 1 indicates perfect isotropy. Code search is evaluated using the Mean Reciprocal Rank (MRR) based on the cosine distance between the comments and codes. For each programming language, we rank all codes for each comment in the test set.

4 Results

To evaluate the embedding space of the models, we first measure their isotropy and ranking performance. To better assess the difference between natural and programming language, isotropy is measured separately for code and comment representations. Table 2 presents the MRR and IsoScores.

In summary, CodeT5+ achieves the highest MRR and IsoScores, surpassing even fine-tuned (FT) CodeBERT models. While fine-tuning greatly improves CodeBERT's ranking performance, its impact on isotropy is minor, with an average IsoScore increase of 0.073. Supplementary experiments with disabling the down-projection in CodeT5+ show worse results than fine-tuning CodeBERT, demonstrating that using a smaller hidden dimension in itself benefits both isotropy and ranking performance. Overall, our analysis shows that while models with higher isotropy perform better, the relationship between MRR and IsoScore is not linear. Additionally, the analysis reveals that the isotropy of code and comment embeddings differs only marginally. This suggests that code and comment embeddings can be treated as similarly isotropic in practice.

Applying standard ZCA whitening (where $\epsilon = 0$) greatly improves the base CodeBERT and Code Llama results, but in the case of fine-tuned CodeBERT and CodeT5+, it decreased the ranking performance on most datasets. With the introduction of the eigenvalue regularizer, we found that moderate whitening ($\epsilon \in \{0.1, 0.01\}$) results in the best performance with the base models, and only the fine-tuned CodeBERT requires stronger whitening($\epsilon = 0.0001$) for optimal performance. Figure 1 showcases the interaction between the eigenvalue regularizer and Isoscore/MRR. Overall, we find that the optimal IsoScore for the base models ranges between 0.2 and 0.8, while the fine-tuned model performs best with almost perfect isotropy. This pattern is demonstrated in Table 3, where fine-tuned CodeBERT achieves IsoScores consistently above 0.99 across all programming languages while delivering moderate MRR improvements (ranging from+0.042 to +0.075), whereas Code Llama shows more substantial MRR gains (up to +0.476 for Ruby) with IsoScores between 0.224 and 0.496. Importantly, the positive Δ MRR values across nearly all models and programming languages demonstrate that the Soft-ZCA whitening technique can reliably improve code search performance. This suggests the technique is robust and effective across different model architectures and provides performance benefits even for low-resource programming languages not present in the training data.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

Table 2: MRR and IsoScores (Code / Comment) on the CodeSearchNet and StatCodeSearch(R) datasets using non-whitened embeddings.

	CodeBERT		FT CodeBERT		CodeT5+		Code LLama	
	MRR	IsoScores	MRR	IsoScores	MRR	IsoScores	MRR	IsoScores
Ruby	0.006	0.007 / 0.014	0.547	0.052 / 0.062	0.705	0.350 / 0.296	0.047	0.008 / 0.003
Javascript	0.002	0.005 / 0.013	0.427	0.065 / 0.072	0.638	0.365 / 0.335	0.026	0.010 / 0.002
Go	0.002	0.006 / 0.010	0.619	0.050 / 0.036	0.757	0.234 / 0.196	0.031	0.006 / 0.003
Java	0.000	0.007 / 0.007	0.395	0.059 / 0.067	0.595	0.388 / 0.313	0.015	0.009 / 0.002
Python	0.001	0.006 / 0.021	0.500	0.067 / 0.071	0.721	0.394 / 0.356	0.017	0.007 / 0.005
PHP	0.000	0.006 / 0.007	0.248	0.072 / 0.048	0.537	0.400 / 0.262	0.009	0.009 / 0.001
R	0.011	0.005 / 0.004	(no fine-tuning data)		0.045	0.139 / 0.118	0.024	0.002 / 0.002

Table 3: MRR improvement as difference to non-whitened embeddings and IsoScores (Code / Comment) of the whitened embeddings using the best epsilon

	CodeBERT		FT CodeBERT		CodeT5+		Code LLama	
	ΔMRR	IsoScores	ΔMRR	IsoScores	Δ MRR	IsoScores	ΔMRR	IsoScores
Ruby	+0.230	0.365 / 0.511	+0.075	0.998 / 0.998	+0.007	0.377 / 0.340	+0.476	0.224 / 0.298
Javascript	+0.142	0.348 / 0.551	+0.049	0.992 / 0.994	+0.003	0.394 / 0.367	+0.369	0.299 / 0.428
Go	+0.250	0.673 / 0.863	+0.042	0.998 / 0.998	0.000	0.317 / 0.291	+0.465	0.257 / 0.433
Java	+0.148	0.736 / 0.889	+0.064	0.991 / 0.992	+0.002	0.522 / 0.495	+0.329	0.453 / 0.388
Python	+0.156	0.381 / 0.555	+0.062	0.998 / 0.998	0.000	0.420 / 0.386	+0.399	0.326 / 0.496
PHP	+0.102	0.726 / 0.899	+0.055	0.998 / 0.998	+0.002	0.423 / 0.327	+0.227	0.275 / 0.450
R	+0.077	0.462 / 0.352	(no fine-tuning data)		+0.035	0.706 / 0.641	+0.337	0.247 / 0.248

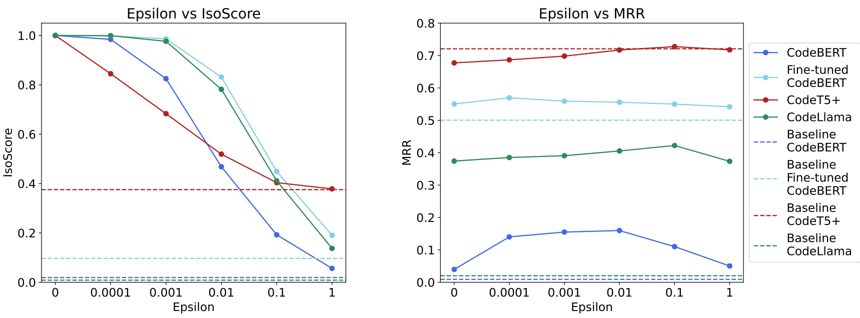

Fig. 1: Average IsoScore (left) and MRR measures (right) at different epsilon values on the CodeSearchNet Python dataset

5 Conclusion

Controlling isotropy through Soft-ZCA whitening offers a simple yet effective way to improve code search performance across different code LMs and programming languages. The consistent MRR improvements suggest that embedding space geometry plays a crucial role in semantic code search. By improving the isotropy of embeddings, this post-processing technique offers a practical solution for enhancing code search systems in production.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

References

[1] X. Cai, J. Huang, Y. Bian, and K. Church. Isotropy in the contextual embedding space: Clusters and manifolds. In *ICLR*, 2020.

[2] J. Attieh, A. Woubie Zewoudie, V. Vlassov, A. Flanagan, and T. Bäckström. Optimizing the performance of text classification models by improving the isotropy of the embeddings using a joint loss function. In *ICDAR*, 2023.

[3] Y. Ji, J. Wang, J. Li, H. Ye, and M. Zhang. Isotropic representation can improve zero-shot cross-lingual transfer on multilingual language models. In *EMNLP*, 2023.

[4] H. Husain, H. Wu, T. Gazit, M. Allamanis, and M. Brockschmidt. CodeSearchNet challenge: Evaluating the state of semantic code search. *arXiv:1909.09436*, 2019.

[5] A. Eghbali and M. Pradel. CrystalBLEU: precisely and efficiently measuring the similarity of code. In *International Conference on Automated Software Engineering*, 2022.

[6] Y. Wang, H. Le, A. Gotmare, N. Bui, et al. CodeT5+: Open code large language models for code understanding and generation. In *EMNLP*, 2023.

[7] A. Diera, A. Dahou, L. Galke, F. Karl, et al. GenCodeSearchNet: A benchmark test suite for evaluating generalization in programming language understanding. In *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*, 2023.

[8] S. Rajaee and M. T. Pilehvar. How does fine-tuning affect the geometry of embedding space: A case study on isotropy. In *EMNLP*, 2021.

[9] J. Su, J. Cao, W. Liu, and Y. Ou. Whitening sentence representations for better semantics and faster retrieval. *arXiv:2103.15316*, 2021.

[10] A. Bell and T. J. Sejnowski. Edges are the'independent components' of natural scenes. *NIPS*, 1996.

[11] Z. Feng, D. Guo, D. Tang, N. Duan, et al. CodeBERT: A pre-trained model for programming and natural languages. In *EMNLP*, 2020.

[12] B. Roziere, J. Gehring, F. Gloeckle, S. Sootla, et al. Code Llama: Open foundation models for code. *arXiv:2308.12950*, 2023.

[13] A. Kessy, A. Lewin, and K. Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 72(4):309–314, 2018.

[14] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018.

[15] W. Rudman, N. Gillman, T. Rayne, and C. Eickhoff. IsoScore: Measuring the uniformity of embedding space utilization. In *ACL*, 2022.