Appendix

Anonymous Author(s)

I. Baselines

To evaluate the effectiveness of our approach, we compare CoCoSoDa with four deep end-to-end approaches and ten pre-training-based approaches,RoBERTa [1], RoBERTa (code) [2], CodeBERT [2], GraphCodeBERT [3], Corder [4], CodeT5 [5], PLBART [6], SPT-Code [7], SyncoBERT [8], UniXcoder [9], Among them, Corder, SyncoBERT and UniXcoder adopt some constrastive learning-related techniques to pre-train the source code model.

- **NBow**, **CNN**, **BiRNN** and **SelfAtnn** [10] use various encoding models such as neural bag-of-words [11], 1D convolultional neural network [12] , bi-directional GRU [13], and multi-head attention [14] to obtain the representations of code snippets and queries. Then they use inner product of the representations of code snippets and queries to measure their semantic similarity.
- **RoBERTa** [1] and **RoBERTa (code)** [2] are built on a multi-layer bidirectional Transformer [14] encoder and pre-trained with masked language modeling (MLM) task, which is to predict the original tokens of the masked positions. The formal is pre-trained on natural language corpus [1], while the latter is pre-trained on source code corpus [10].
- **CodeBERT** [2] is built on the same architecture of RoBERTa and is pre-trained with large-scale code-text pairs on two tasks: MLM and replaced token detection (RTD), which uses a discriminator to identify the replaced token.
- **GraphCodeBERT** [3] considers the code structure information for pre-training and achieves the state-of-art results among code search baselines. The pre-training tasks include MLM, data flow edge prediction, and node alignment.
- **Corder** [4] firstly uses an unimodal contrastive learning approach (only the code modality) to pre-train the model and fine-tune it in the downstream tasks. To generate the functional equivalence code snippets. As Corder does not release the implementation of semantic-preserving transformations and it is costly to implement these transformations for five programming languages, we only implemented for Java language because Java is the most studied language for code search [15]. To make a fair comparison, we use the unimodal contrastive learning technique to continually pre-train the UniXcoder and then fine-tune it on the code search task as the implementation of Corder.
- **CodeT5** [5], **PLBART** [6], **SPT-Code** [7] are sequence-to-sequence pre-trained models for source code. The first is pre-trained with three identifier-aware pre-training tasks to enable the model to identify identifiers in source code or recover masked identifiers. The second is pre-trained on a large-scale of Java and Python corpus with denoising autoencoding, which is to reconstruct the input code sequence that is corrupted by a one of noise strategies. These strategies include token masking, token deletion and token infilling, which mask a span of tokens.The third takes source code and corresponding AST and paired summarization as input and is pre-trained with three code-specific tasks.
- **SyncoBERT** [8] are syntax-guided multi-modal contrastive pre-training for code representation. It takes source code, AST and summarization as input and pre-trained with identifier prediction and AST edge prediction to learn the lexical and syntactic knowledge of source code. In addition, they treat the input as three different modalities and propose a multi-modal contrastive learning approach to maximize the mutual information among different modalities.
- **UniXcoder** [9] takes two-modality data, paired summarization and simplified AST of source code, as input. To improve the understanding of source code, the authors pre-train the model with MLM, unidirectional language modeling (ULM), denoising autoencoder, and two contrastive learning-related tasks.

In our experiments, we train the four deep end-to-end approaches from scratch, and for the ten pre-trained approaches, we initialize them with the pre-trained models and fine-tune (or continually pre-train and fine-tune) them according to the descriptions in their original papers or their released source code.

II. Experimental Results

A. RQ1: What is the Effectiveness of CoCoSoDa?

We evaluate the effectiveness of our model CoCoSoDa by comparing it to some baselines on the CodeSearchNet dataset with six programming languages. We present the results under R@1, R@5 and R@10 here. The results are shown in Table I, II and III . We can see that CoCoSoDa performs best among all approaches.

B. RQ2: How much do Different Components Contribute?

In this section, we study the contribution of each component of our approach CoCoSoDa. The experimental results are shown in Table IV, V and VI. We can see that the performance of the model drops after removing any one component. It demonstrates that each component plays an important role in the code search model.

TABLE I
THE PERFORMANCE OF DIFFERENT APPROACHES ON R@1. JS IS SHORT FOR JAVASCRIPT. COCOSODA OUTPERFORMS SIGNIFICANTLY (STATISTICAL SIGNIFICANCE $p < 0.01$).

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
RoBERTa	0.469	0.413	0.8	0.48	0.499	0.45	0.519
RoBERTa	0.524	0.452	0.811	0.511	0.528	0.467	0.549
CodeBERT	0.583	0.514	0.837	0.574	0.58	0.521	0.602
GraphCodeBERT	0.607	0.538	0.858	0.594	0.592	0.545	0.622
Corder	-	-	-	-	0.637	-	-
PLBART	0.582	0.511	0.838	0.567	0.566	0.509	0.596
CodeT5	0.621	0.556	0.839	0.599	0.585	0.541	0.624
SyncoBERT	0.622	0.584	0.871	0.616	0.615	0.575	0.647
UniXcoder	0.642	0.591	0.875	0.612	0.618	0.572	0.652
SPT-Code	0.605	0.535	0.847	0.599	0.603	0.546	0.622
CoCoSoDa	**0.719**	**0.661**	**0.880**	**0.665**	**0.618**	**0.598**	**0.690**

TABLE II
THE PERFORMANCE OF DIFFERENT APPROACHES ON R@5. JS IS SHORT FOR JAVASCRIPT. COCOSODA OUTPERFORMS SIGNIFICANTLY (STATISTICAL SIGNIFICANCE $p < 0.01$).

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
RoBERTa	0.717	0.652	0.926	0.727	0.737	0.694	0.742
RoBERTa (code)	0.761	0.716	0.93	0.756	0.77	0.715	0.775
CodeBERT	0.800	0.752	0.944	0.792	0.796	0.753	0.806
GraphCodeBERT	0.824	0.774	0.954	0.813	0.817	0.785	0.835
Corder	-	-	-	-	0.841	-	-
PLBART	0.799	0.756	0.945	0.783	0.786	0.752	0.804
CodeT5	0.839	0.779	0.946	0.783	0.805	0.781	0.822
SyncoBERT	0.85	0.795	0.964	0.842	0.841	0.801	0.849
UniXcoder	0.86	0.804	0.966	0.838	0.853	0.8	0.853
SPT-Code	0.822	0.77	0.953	0.784	0.821	0.785	0.822
CoCoSoDa	**0.947**	**0.896**	**0.971**	**0.888**	**0.853**	**0.834**	**0.898**

C. RQ5: Why does Our Model CoCoSoDa Work?

1) Qualitative analysis: We also visualize learned representations to help intuitively understand why CoCoSoDa works well. Specifically, firstly, we randomly sample X (X=100,200,300,or 400) code-query pairs from Java dataset with ten different random seeds from 0 to 9. The following findings and conclusions hold for different X and random seeds. Second, we feed the sampled data including code snippets and queries to well-trained CoCoSoDa and UniXcoder individually and obtain the their representations. Third, we apply T-SNE [16] to reduce the dimensionality of obtained representations into 2D and visualize dimensionality-reduced representations in Fig. 1 to 40. We can see both CoCoSoDa and UniXcoder can map most paired code snippets and queries into close embeddings. *(2)* UniXcoder has more cases than CoCoSoDa, where pairwise representations of queries and code snippets are far away from each other. In the future, we will conduct error analysis to study the paired samples with long green lines. In summary, the visualization results intuitively show that CoCoSoDa can learn the better representations than UniXcoder.

TABLE III
THE PERFORMANCE OF DIFFERENT APPROACHES ON R@10. JS IS SHORT FOR JAVASCRIPT. COCOSODA OUTPERFORMS SIGNIFICANTLY (STATISTICAL SIGNIFICANCE $p < 0.01$).

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
RoBERTa	0.785	0.730	0.949	0.793	0.796	0.764	0.803
RoBERTa (code)	0.821	0.794	0.952	0.819	0.831	0.783	0.833
CodeBERT	0.853	0.814	0.962	0.850	0.852	0.814	0.857
GraphCodeBERT	0.872	0.834	0.972	0.866	0.865	0.832	0.865
Corder	-	-	-	-	0.884	-	-
PLBART	0.851	0.819	0.963	0.841	0.84	0.816	0.855
CodeT5	0.883	0.838	0.964	0.869	0.86	0.827	0.874
SyncoBERT	0.868	0.847	0.979	0.893	0.891	0.865	0.89
UniXcoder	0.898	0.854	0.98	0.889	0.905	0.863	0.898
SPT-Code	0.87	0.831	0.974	0.872	0.87	0.833	0.875
CoCoSoDa	**0.973**	**0.942**	**0.983**	**0.930**	**0.905**	**0.897**	**0.938**

TABLE IV
ABLATION STUDY OF COCOSODA ON R@1.

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CoCoSoDa	**0.719**	**0.661**	**0.880**	**0.665**	**0.618**	**0.598**	**0.690**
w/o DR	0.700	0.609	0.856	0.627	0.649	0.584	0.671
w/o DM	0.696	0.649	0.858	0.640	0.658	0.583	0.681
w/o DRST	0.705	0.611	0.857	0.627	0.648	0.582	0.672
w/o DMST	0.691	0.613	0.852	0.626	0.651	0.580	0.669
w/o 4-SoDas	0.676	0.613	0.854	0.635	0.642	0.578	0.666
w/o inter-modal loss	0.679	0.603	0.856	0.627	0.644	0.568	0.663
w/o intra-modal loss	0.686	0.603	0.853	0.622	0.650	0.575	0.665

III. DISCUSSION

A. What is the Effectiveness of CoCoSoDa without being Fine Tuned?

To further study the effectiveness of CoCoSoDa, we evaluate different pre-trained models under the zero-short experimental setting, where models are directly used for evaluation without fine-tuning them. The experimental results are shown in Table VII, VII and VII. Our pre-trained model performs better than other pre-trained models when all of them are not fine-tuned. Furthermore, the performance of CoCoSoDa without being fine-tuned is even better than many fine-tuned models.

REFERENCES

[1] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019.

[2] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, and M. Zhou, "Codebert: A pre-trained model for programming and natural languages," in *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, ser. Findings of ACL, T. Cohn, Y. He, and Y. Liu, Eds., vol. EMNLP 2020. Association for Computational Linguistics, 2020, pp. 1536–1547. [Online]. Available: https://doi.org/10.18653/v1/2020.findings-emnlp.139

[3] D. Guo, S. Ren, S. Lu, Z. Feng, D. Tang, S. Liu, L. Zhou, N. Duan, A. Svyatkovskiy, S. Fu, M. Tufano, S. K. Deng, C. B. Clement, D. Drain, N. Sundaresan, J. Yin, D. Jiang, and M. Zhou, "Graphcodebert: Pre-training code representations with data flow," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=jLoC4ez43PZ

[4] N. D. Q. Bui, Y. Yu, and L. Jiang, "Self-supervised contrastive learning for code retrieval and summarization via semantic-preserving transformations," in *SIGIR*. ACM, 2021, pp. 511–521.

TABLE V
ABLATION STUDY OF COCOSODA ON R@5.

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CoCoSoDa	**0.947**	**0.896**	**0.971**	**0.888**	**0.853**	**0.834**	**0.898**
w/o DR	0.916	0.845	0.966	0.857	0.863	0.812	0.877
w/o DM	0.933	0.889	0.966	0.873	0.878	0.821	0.893
w/o DRST	0.91	0.84	0.965	0.858	0.862	0.81	0.874
w/o DMST	0.913	0.846	0.964	0.861	0.867	0.815	0.878
w/o 4-SoDas	0.906	0.83	0.96	0.861	0.859	0.815	0.872
w/o inter-modal loss	0.9	0.822	0.966	0.854	0.86	0.803	0.868
w/o intra-modal loss	0.897	0.831	0.963	0.855	0.861	0.809	0.869

TABLE VI
ABLATION STUDY OF COCOSODA ON R@10.

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CoCoSoDa	**0.973**	**0.942**	**0.983**	**0.930**	**0.905**	**0.897**	**0.938**
w/o DR	0.962	0.902	0.979	0.909	0.905	0.875	0.922
w/o DM	0.971	0.932	0.980	0.925	0.921	0.884	0.936
w/o DRST	0.956	0.898	0.980	0.907	0.904	0.874	0.92
w/o DMST	0.961	0.902	0.979	0.911	0.906	0.876	0.922
w/o 4-SoDas	0.945	0.885	0.982	0.911	0.902	0.872	0.916
w/o inter-modal loss	0.942	0.882	0.981	0.904	0.902	0.866	0.913
w/o intra-modal loss	0.940	0.889	0.98	0.905	0.903	0.870	0.915

[5] Y. Wang, W. Wang, S. R. Joty, and S. C. H. Hoi, "Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation," in *EMNLP (1)*. Association for Computational Linguistics, 2021, pp. 8696–8708.

[6] W. U. Ahmad, S. Chakraborty, B. Ray, and K. Chang, "Unified pre-training for program understanding and generation," in *NAACL-HLT*. Association for Computational Linguistics, 2021, pp. 2655–2668.

[7] C. Niu, C. Li, V. Ng, J. Ge, L. Huang, and B. Luo, "Spt-code: Sequence-to-sequence pre-training for learning source code representations," in *ICSE*. ACM, 2022, pp. 1–13.

[8] X. Wang, Y. Wang, F. Mi, P. Zhou, Y. Wan, X. Liu, L. Li, H. Wu, J. Liu, and X. Jiang, "Syncobert: Syntax-guided multi-modal contrastive pre-training for code representation," *arXiv preprint arXiv:2108.04556*, 2021.

[9] D. Guo, S. Lu, N. Duan, Y. Wang, M. Zhou, and J. Yin, "Unixcoder: Unified cross-modal pre-training for code representation," in *ACL (1)*. Association for Computational Linguistics, 2022, pp. 7212–7225.

[10] H. Husain, H. Wu, T. Gazit, M. Allamanis, and M. Brockschmidt, "Codesearchnet challenge: Evaluating the state of semantic code search," *CoRR*, vol. abs/1909.09436, 2019. [Online]. Available: http://arxiv.org/abs/1909.09436

[11] M. Iyyer, V. Manjunatha, J. L. Boyd-Graber, and H. D. III, "Deep unordered composition rivals syntactic methods for text classification," in *ACL (1)*. The Association for Computer Linguistics, 2015, pp. 1681–1691.

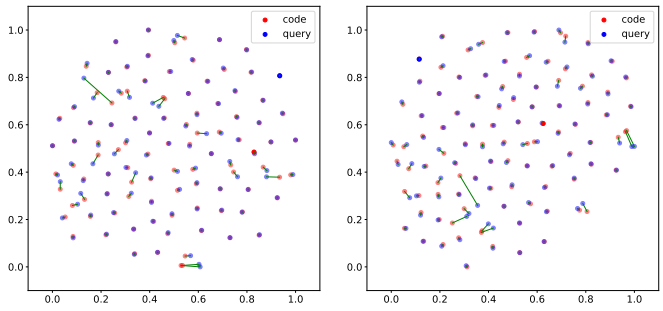
[12] Y. Kim, "Convolutional neural networks for sentence classification," in *EMNLP*. ACL, 2014, pp. 1746–1751.

[13] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP*. ACL, 2014, pp. 1724–1734.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

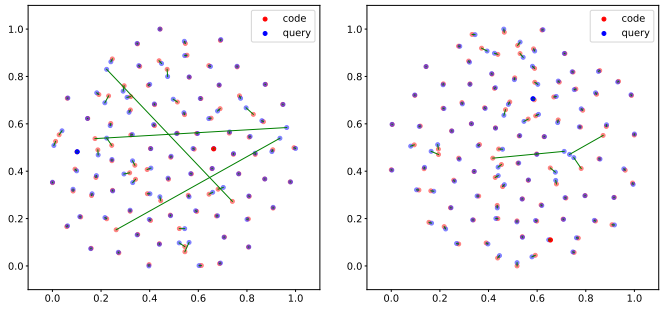
[15] C. Liu, X. Xia, D. Lo, C. Gao, X. Yang, and J. C. Grundy, "Opportunities and challenges in code search tools," *ACM Comput. Surv.*, vol. 54, no. 9, pp. 196:1–196:40, 2022.

[16] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.



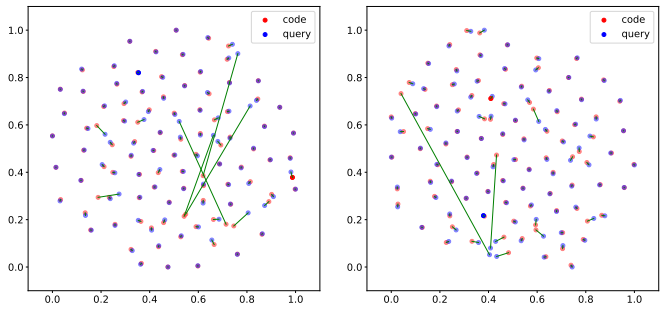
(a) UniXcoder (b) CoCoSoDa

Fig. 1. T-SNE visualization of representations of 100 code-query pairs with random seed of 0. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



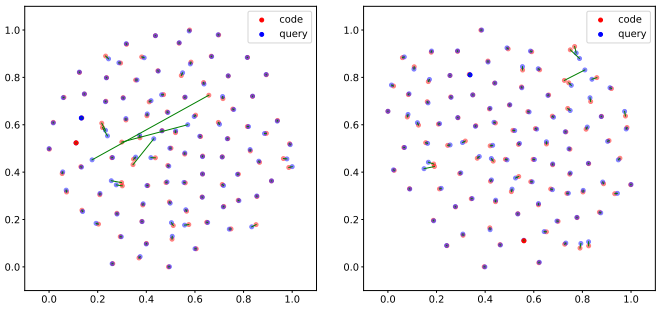
(a) UniXcoder (b) CoCoSoDa

Fig. 2. T-SNE visualization of representations of 100 code-query pairs with random seed of 1. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



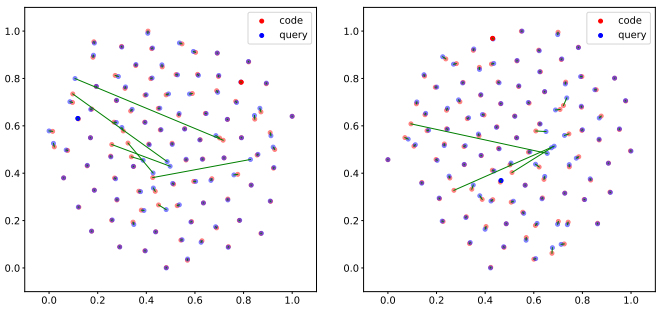
(a) UniXcoder (b) CoCoSoDa

Fig. 3. T-SNE visualization of representations of 100 code-query pairs with random seed of 2. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



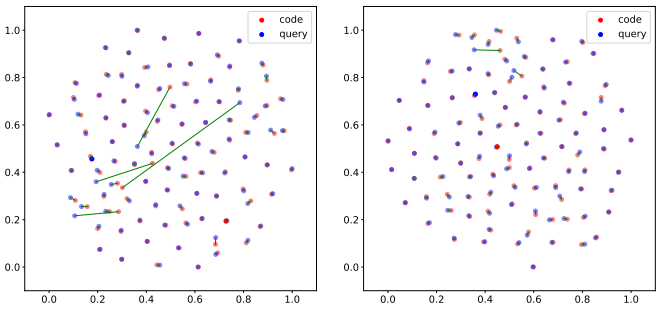
(a) UniXcoder (b) CoCoSoDa

Fig. 4. T-SNE visualization of representations of 100 code-query pairs with random seed of 3. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



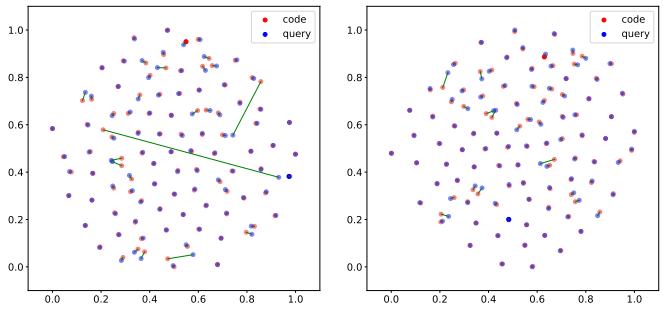
(a) UniXcoder (b) CoCoSoDa

Fig. 5. T-SNE visualization of representations of 100 code-query pairs with random seed of 4. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



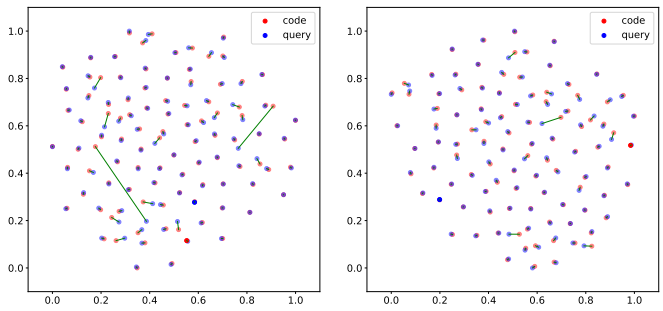
(a) UniXcoder (b) CoCoSoDa

Fig. 6. T-SNE visualization of representations of 100 code-query pairs with random seed of 5. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



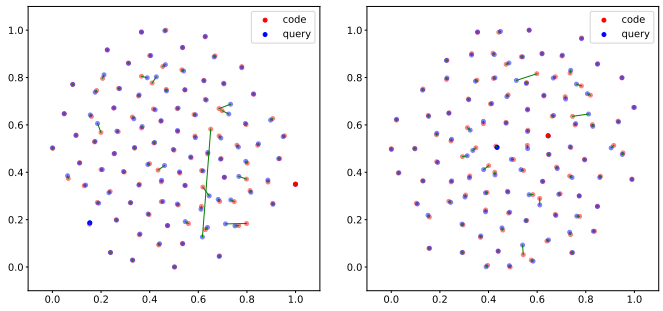
(a) UniXcoder (b) CoCoSoDa

Fig. 7. T-SNE visualization of representations of 100 code-query pairs with random seed of 6. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



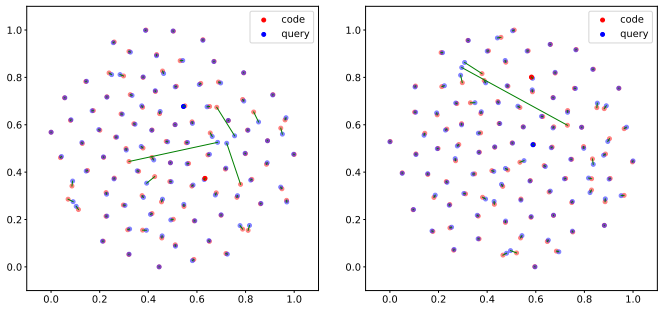
(a) UniXcoder (b) CoCoSoDa

Fig. 8. T-SNE visualization of representations of 100 code-query pairs with random seed of 7. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



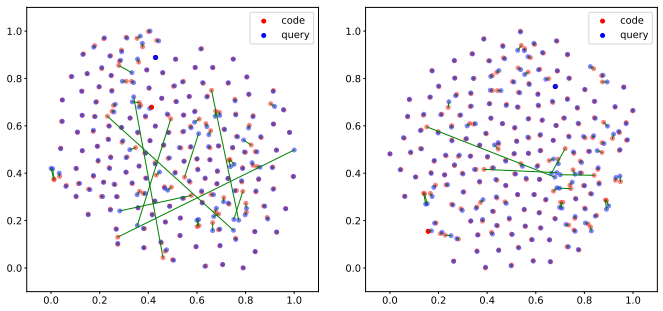
(a) UniXcoder (b) CoCoSoDa

Fig. 9. T-SNE visualization of representations of 100 code-query pairs with random seed of 8. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



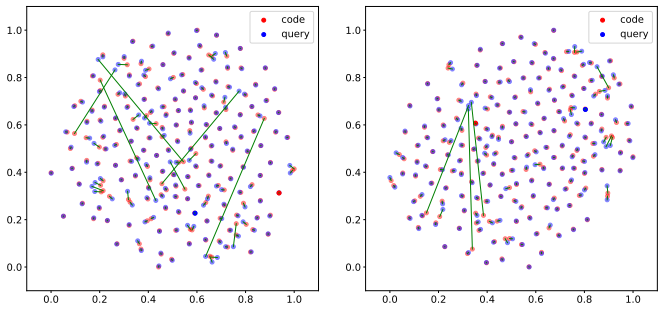
(a) UniXcoder (b) CoCoSoDa

Fig. 10. T-SNE visualization of representations of 100 code-query pairs with random seed of 9. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



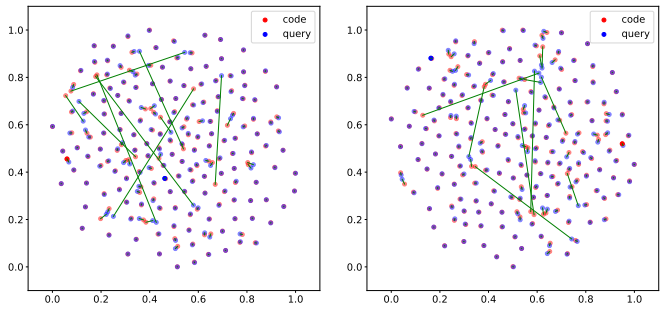
(a) UniXcoder (b) CoCoSoDa

Fig. 11. T-SNE visualization of representations of 200 code-query pairs with random seed of 0. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



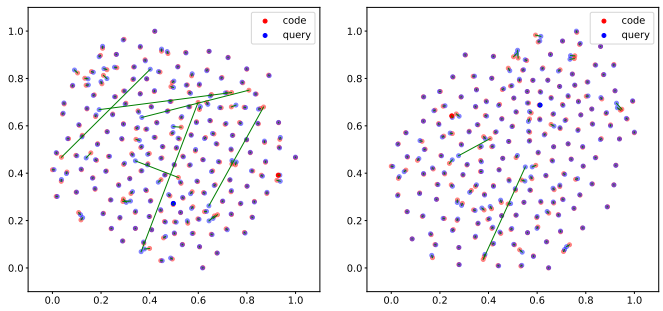
(a) UniXcoder (b) CoCoSoDa

Fig. 12. T-SNE visualization of representations of 200 code-query pairs with random seed of 1. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



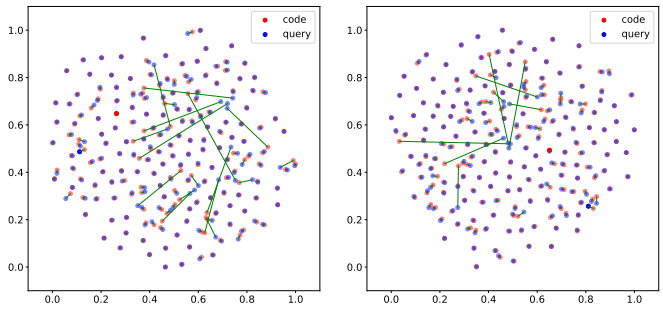
(a) UniXcoder (b) CoCoSoDa

Fig. 13. T-SNE visualization of representations of 200 code-query pairs with random seed of 2. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



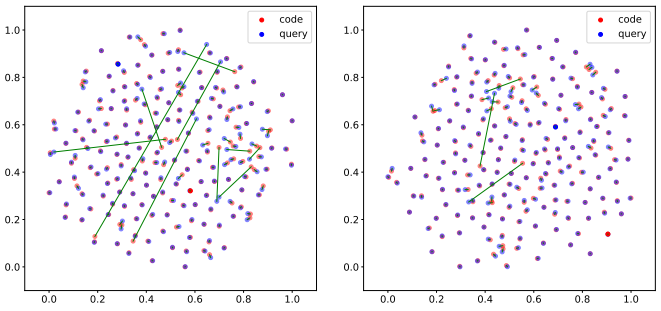
(a) UniXcoder (b) CoCoSoDa

Fig. 14. T-SNE visualization of representations of 200 code-query pairs with random seed of 3. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



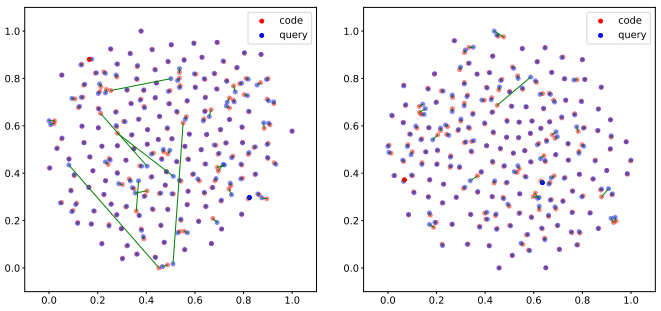
(a) UniXcoder (b) CoCoSoDa

Fig. 15. T-SNE visualization of representations of 200 code-query pairs with random seed of 4. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



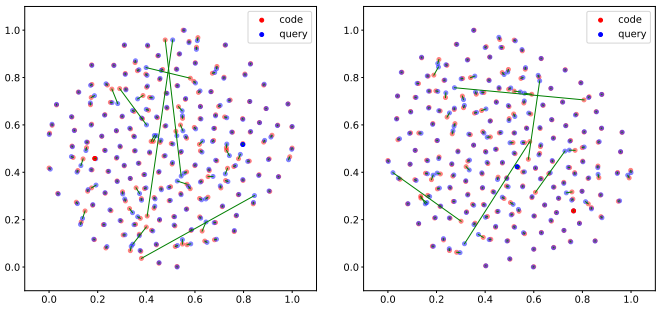
(a) UniXcoder (b) CoCoSoDa

Fig. 16. T-SNE visualization of representations of 200 code-query pairs with random seed of 5. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



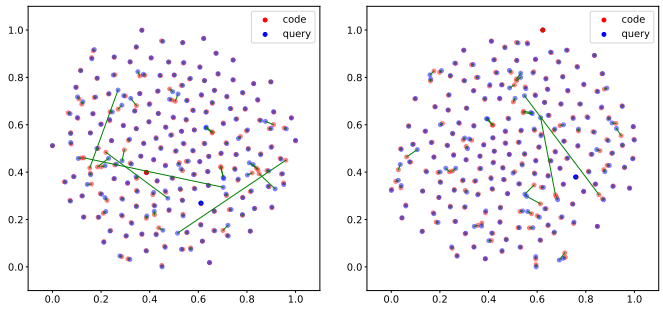
(a) UniXcoder (b) CoCoSoDa

Fig. 17. T-SNE visualization of representations of 200 code-query pairs with random seed of 6. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



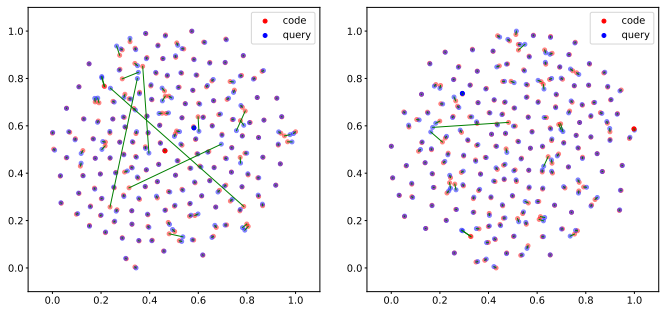
(a) UniXcoder (b) CoCoSoDa

Fig. 18. T-SNE visualization of representations of 200 code-query pairs with random seed of 7. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



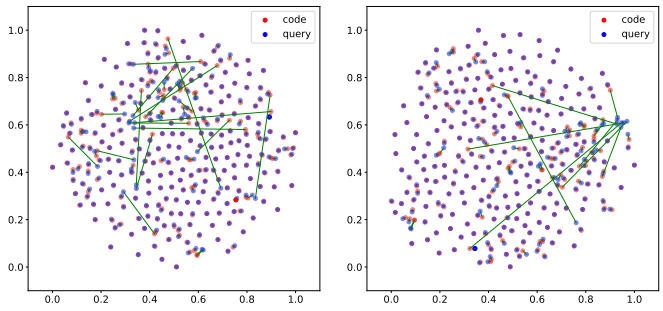
(a) UniXcoder (b) CoCoSoDa

Fig. 19. T-SNE visualization of representations of 200 code-query pairs with random seed of 8. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



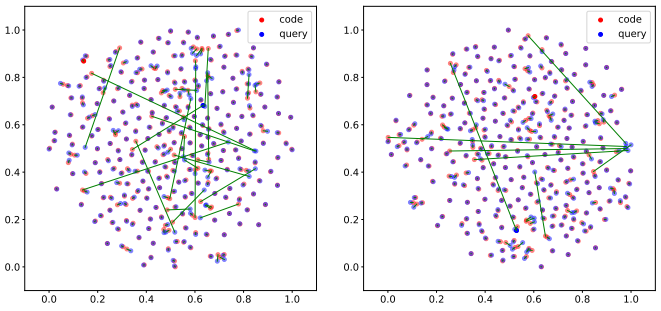
(a) UniXcoder (b) CoCoSoDa

Fig. 20. T-SNE visualization of representations of 200 code-query pairs with random seed of 9. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



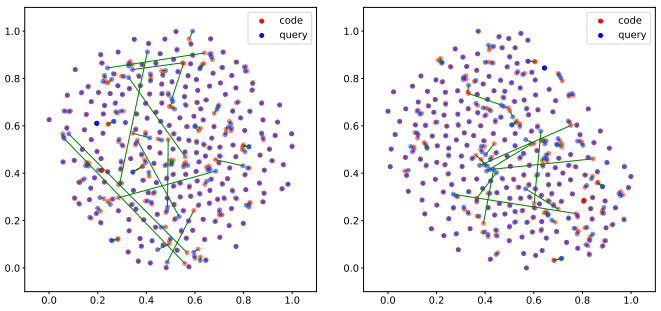
(a) UniXcoder (b) CoCoSoDa

Fig. 21. T-SNE visualization of representations of 300 code-query pairs with random seed of 0. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



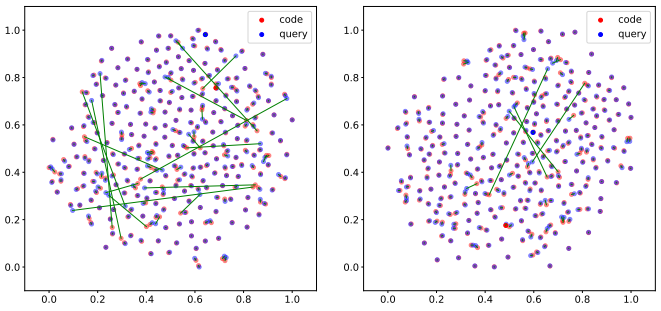
(a) UniXcoder (b) CoCoSoDa

Fig. 22. T-SNE visualization of representations of 300 code-query pairs with random seed of 1. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



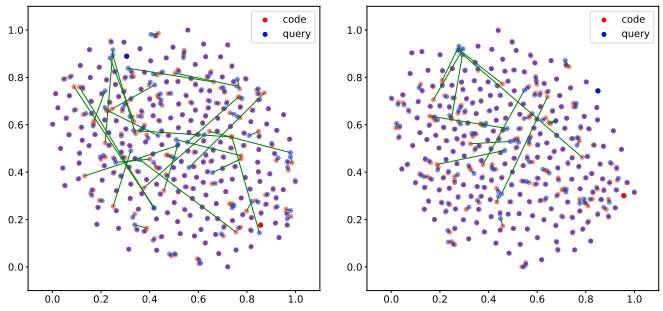
(a) UniXcoder (b) CoCoSoDa

Fig. 23. T-SNE visualization of representations of 300 code-query pairs with random seed of 2. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



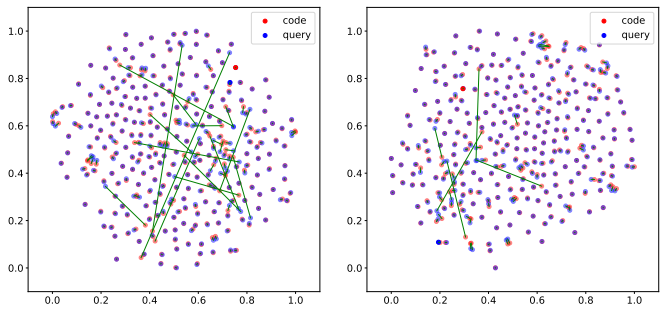
(a) UniXcoder (b) CoCoSoDa

Fig. 24. T-SNE visualization of representations of 300 code-query pairs with random seed of 3. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



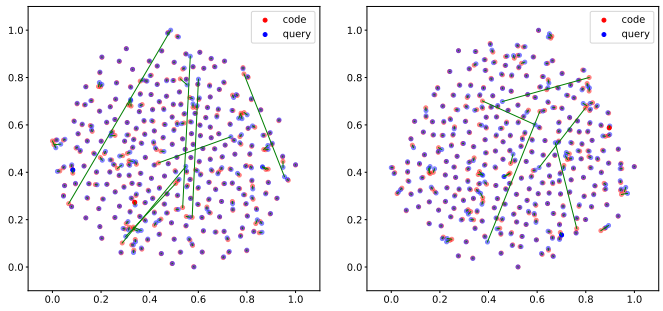
(a) UniXcoder (b) CoCoSoDa

Fig. 25. T-SNE visualization of representations of 300 code-query pairs with random seed of 4. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



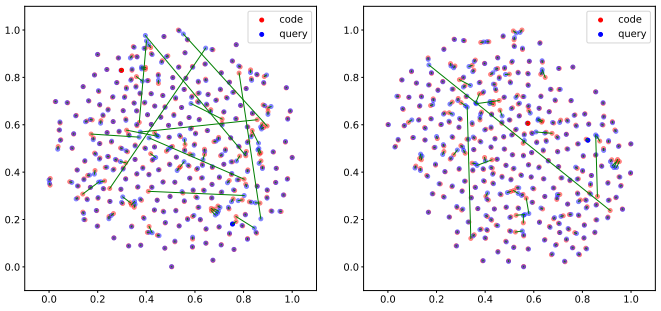
(a) UniXcoder (b) CoCoSoDa

Fig. 26. T-SNE visualization of representations of 300 code-query pairs with random seed of 5. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



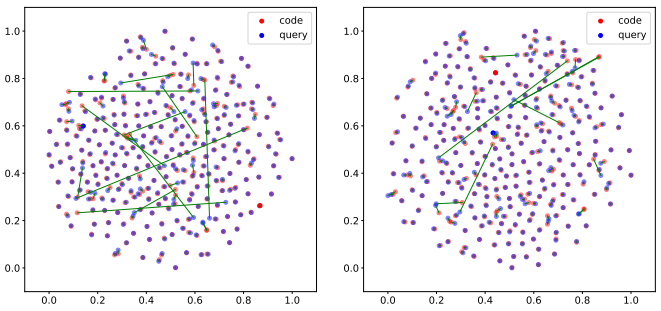
(a) UniXcoder (b) CoCoSoDa

Fig. 27. T-SNE visualization of representations of 300 code-query pairs with random seed of 6. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



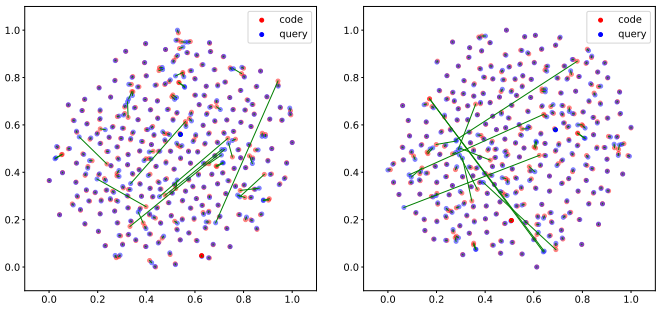
(a) UniXcoder (b) CoCoSoDa

Fig. 28. T-SNE visualization of representations of 300 code-query pairs with random seed of 7. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



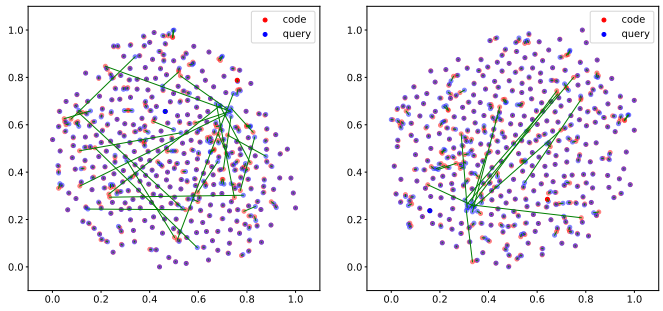
(a) UniXcoder (b) CoCoSoDa

Fig. 29. T-SNE visualization of representations of 300 code-query pairs with random seed of 8. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



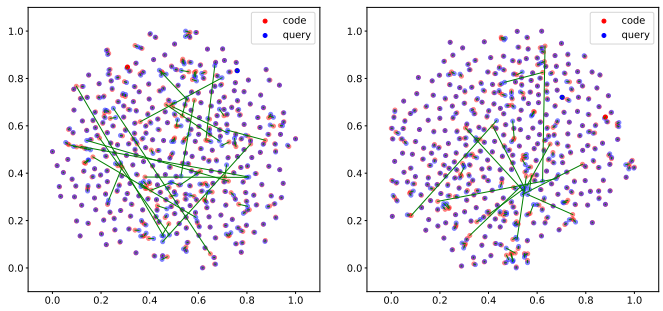
(a) UniXcoder (b) CoCoSoDa

Fig. 30. T-SNE visualization of representations of 300 code-query pairs with random seed of 9. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



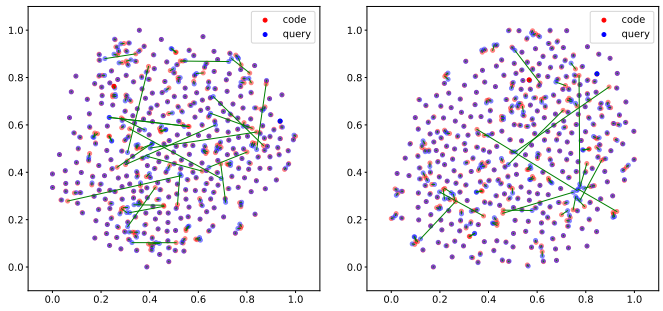
(a) UniXcoder (b) CoCoSoDa

Fig. 31. T-SNE visualization of representations of 400 code-query pairs with random seed of 0. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



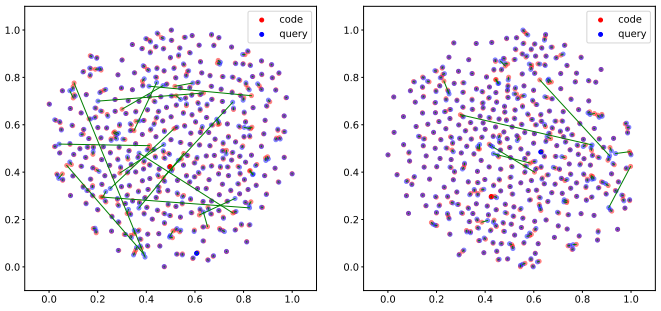
(a) UniXcoder (b) CoCoSoDa

Fig. 32. T-SNE visualization of representations of 400 code-query pairs with random seed of 1. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



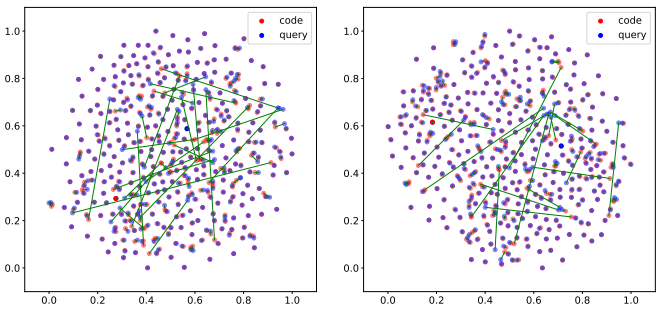
(a) UniXcoder (b) CoCoSoDa

Fig. 33. T-SNE visualization of representations of 400 code-query pairs with random seed of 2. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



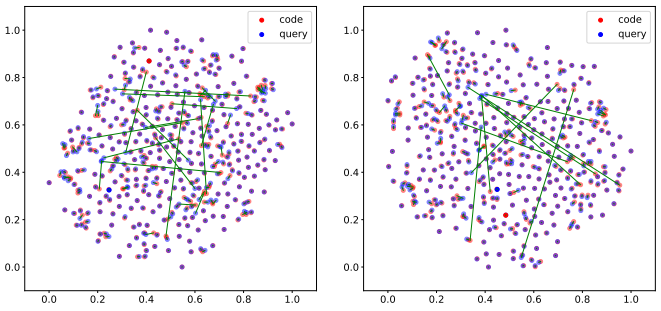
(a) UniXcoder (b) CoCoSoDa

Fig. 34. T-SNE visualization of representations of 400 code-query pairs with random seed of 3. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



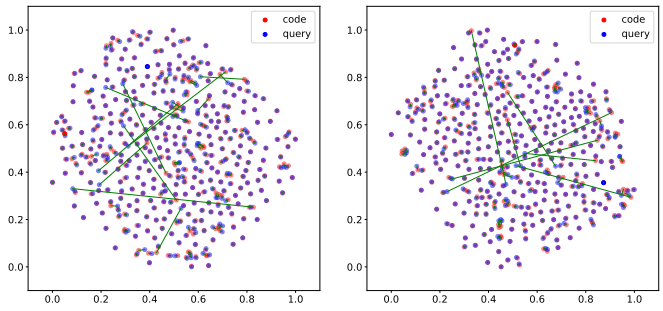
(a) UniXcoder (b) CoCoSoDa

Fig. 35. T-SNE visualization of representations of 400 code-query pairs with random seed of 4. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



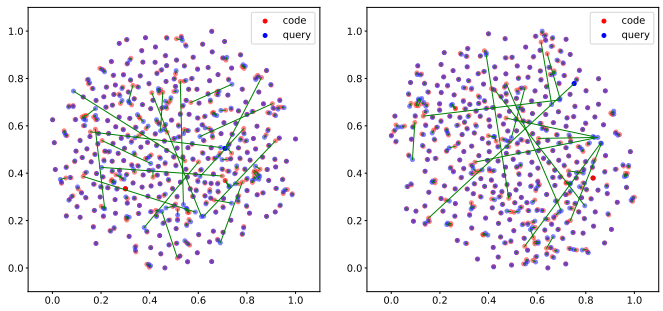
(a) UniXcoder (b) CoCoSoDa

Fig. 36. T-SNE visualization of representations of 400 code-query pairs with random seed of 5. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



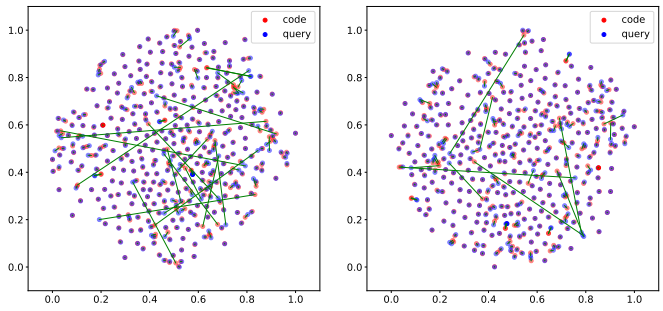
(a) UniXcoder (b) CoCoSoDa

Fig. 37. T-SNE visualization of representations of 400 code-query pairs with random seed of 6. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



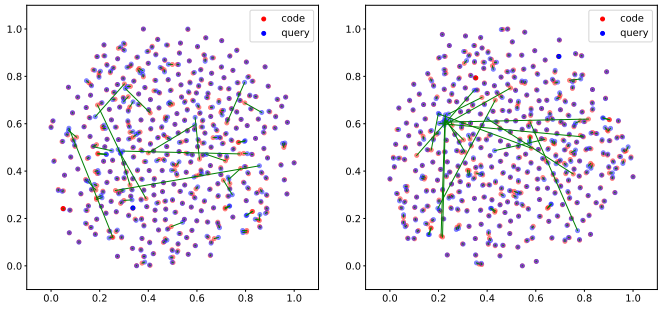
(a) UniXcoder (b) CoCoSoDa

Fig. 38. T-SNE visualization of representations of 400 code-query pairs with random seed of 7. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



(a) UniXcoder (b) CoCoSoDa

Fig. 39. T-SNE visualization of representations of 400 code-query pairs with random seed of 8. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.



(a) UniXcoder (b) CoCoSoDa

Fig. 40. T-SNE visualization of representations of 400 code-query pairs with random seed of 9. (a) and (b) are representations is learned by UniXcoder and CoCoSoDa, respectively. Code is in orange and query is in blue. The distance of paired code and query is indicated by a green line.

TABLE VII
THE PERFORMANCE OF DIFFERENT APPROACHES UNDER THE ZERO-SHORT EXPERIMENTAL SETTING ON R@1

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CodeBERT	0.001	0.001	0.001	0.001	0.001	0.001	0.001
GraphCodeBERT	0.168	0.065	0.139	0.09	0.081	0.076	0.103
UniXcoder	0.473	0.343	0.549	0.344	0.36	0.278	0.391
CodeT5	0.002	0.001	0.005	0.001	0.001	0.001	0.002
CoCoSoDa	**0.684**	**0.6**	**0.821**	**0.589**	**0.612**	**0.528**	**0.639**

TABLE VIII
THE PERFORMANCE OF DIFFERENT APPROACHES UNDER THE ZERO-SHORT EXPERIMENTAL SETTING ON R@5

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CodeBERT	0.002	0.001	0.003	0.001	0.001	0.001	0.002
GraphCodeBERT	0.303	0.148	0.277	0.182	0.159	0.159	0.205
UniXcoder	0.697	0.556	0.767	0.563	0.587	0.477	0.608
CodeT5	0.006	0.004	0.009	0.002	0.001	0.001	0.004
CoCoSoDa	**0.917**	**0.847**	**0.955**	**0.828**	**0.848**	**0.777**	**0.862**

TABLE IX
THE PERFORMANCE OF DIFFERENT APPROACHES UNDER THE ZERO-SHORT EXPERIMENTAL SETTING ON R@10

Model	Ruby	JS	Go	Python	Java	PHP	Avg.
CodeBERT	0.003	0.001	0.006	0.001	0.001	0.001	0.002
GraphCodeBERT	0.375	0.201	0.342	0.227	0.206	0.205	0.259
UniXcoder	0.763	0.623	0.826	0.641	0.662	0.556	0.678
CodeT5	0.010	0.005	0.014	0.002	0.002	0.001	0.006
CoCoSoDa	**0.960**	**0.910**	**0.975**	**0.889**	**0.905**	**0.850**	**0.915**