JPMorgan Chase & Co.

CV4Code: Sourcecode Understanding via Visual Code Representations

Ruibo Shi, Lili Tao, Rohan Saphal, Fran Silavong, Sean Moran

CTO, JP Morgan Chase

Introduction

- Machine Learning on Sourcecode (MLOnCode) promises to redefine how software is delivered through intelligent augmentation of the software development lifecycle (SDLC).
- Core to the field of MLOnCode is the learning of sourcecode feature representations ("code vectors"). Popular methods include code2vec [1] and transformer architectures [8] that capture structure and context.
- We represent sourcecode in a visual way as images that explicitly, through the unique 2D representation, present both the code structure and context directly to the learning algorithm.
- CV4Code is a novel and compact encoding of sourcecode as a 2D spatial grid of numeric values that represent the characters in the code by their ASCII codepoints.
- CV4Code code vectors can be used as code embeddings for MLOnCode tasks, similar to VGG features [7] that have been shown to be a powerful and flexible embedding of images for computer vision tasks.

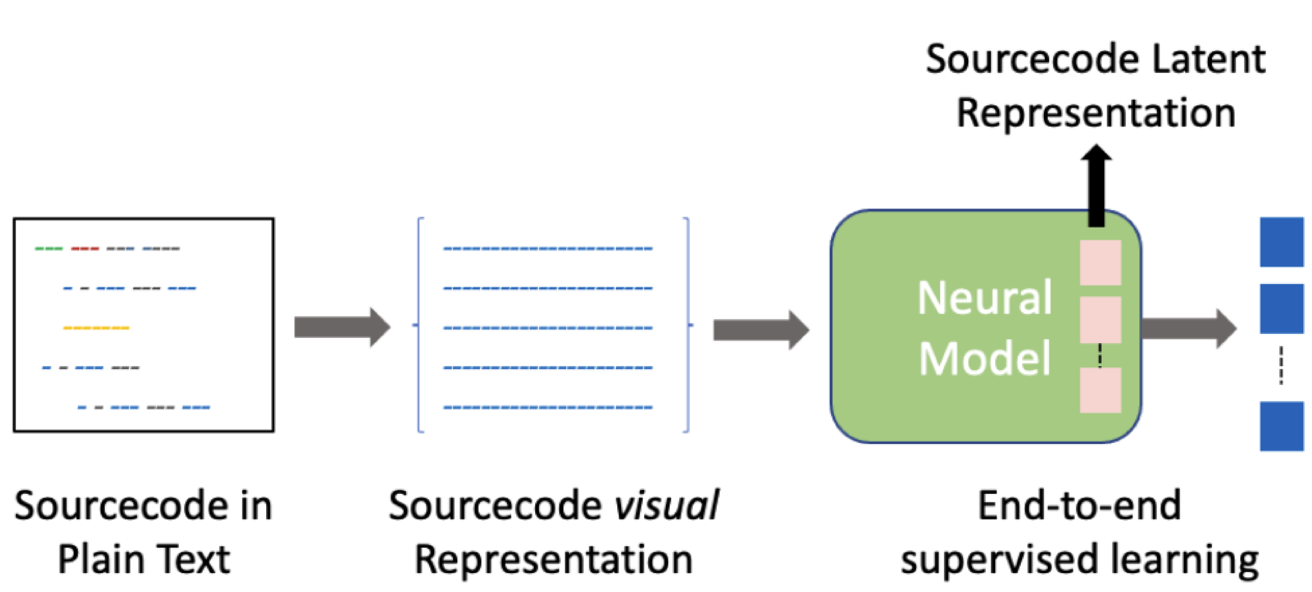


Figure 1: The proposed CV4Code code understanding pipeline

Representing Code as Images

- **CV4Code**: Code snippets are transformed into 2-dimensional (matrix) representation by mapping each printable ASCII character to their unique index values and padding the special [blank] token wherever necessary to retain the rectangular shape of the output.
- Figure 2 shows an example of the code representation generation process. For a code snippet spanning L lines each with C_l, $l \in 0, ..., L-1$ characters, the transformation is done in three steps:
  1. Remove characters not within the valid set, output has $\hat{L}$ lines each with $\hat{C}_l$, $l \in 0, ..., \hat{L}-1$ characters;
  2. Map each input character $v_k \in \mathbb{V}_c$ to its index value k;
  3. Pad each line to $M = \max_{l=0}^{\hat{L}-1} \hat{C}_l$ long with the index value of *[blank]*, generate the output 2-dimensional code matrix $X \in \mathbb{R}^{L \times M}$.

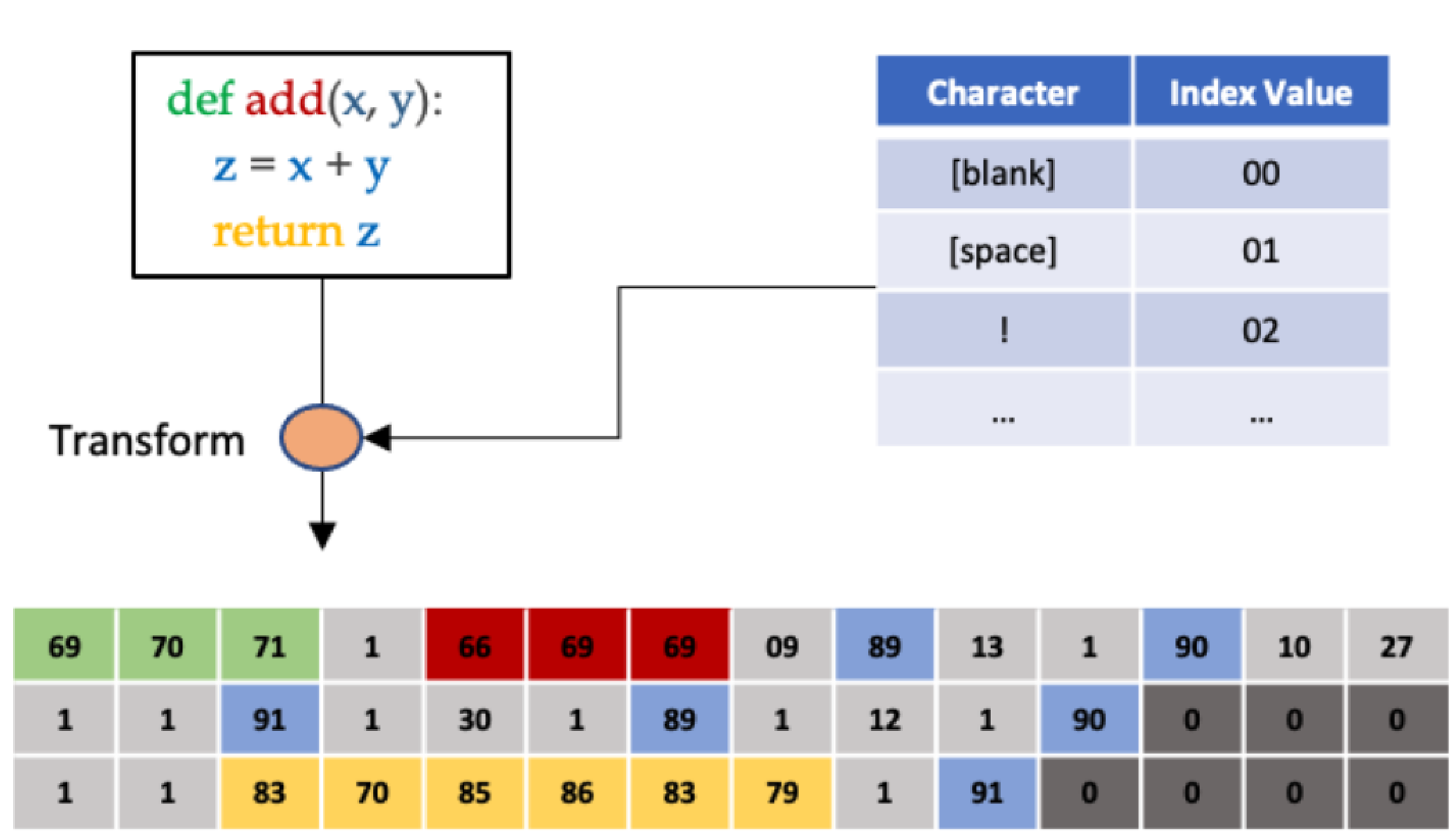
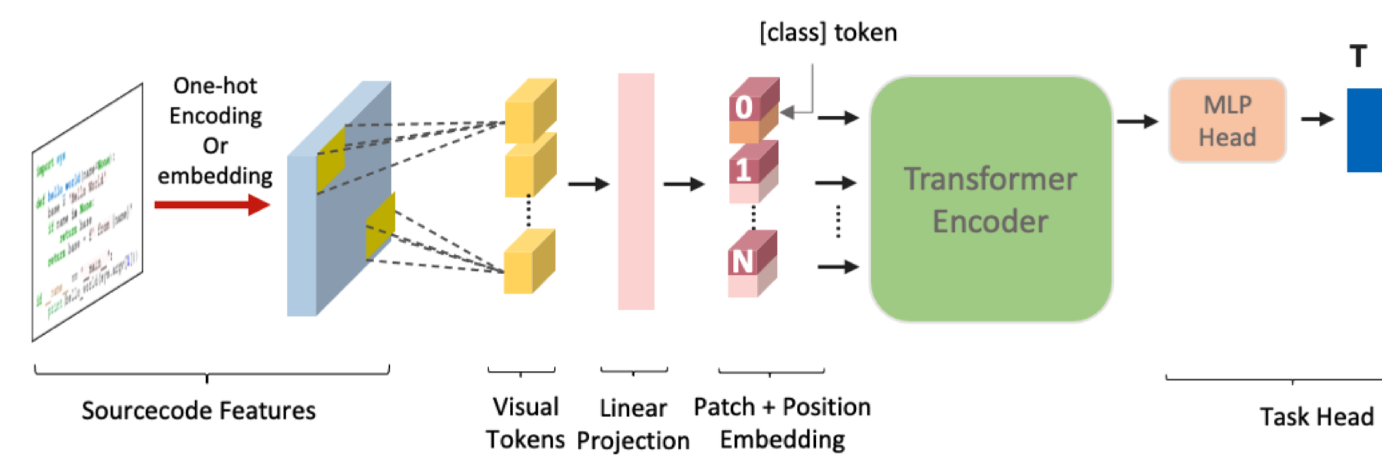
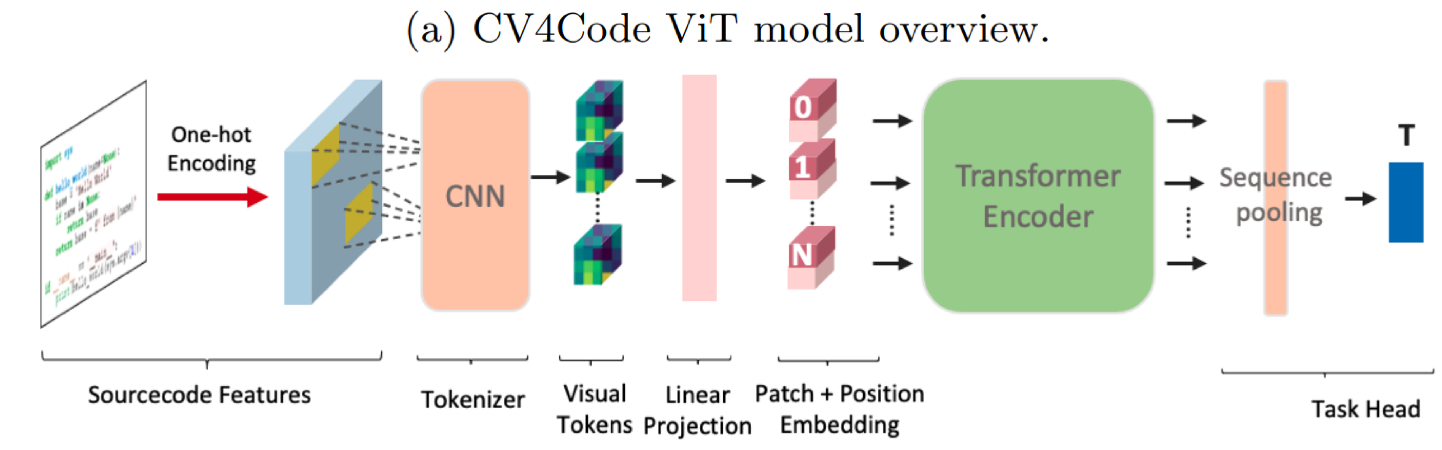


Figure 2: Example of 2D code representation generation.

Models



(a) CV4Code ViT model overview.



(b) CV4Code CCT model overview.

Figure 3: CV4Code transformer model variants.

- Models: ResNet [3], ViT [4], ViT for small-size datasets (ViT-fsd) [5] and hybrid Convolutional Transformer (Conv-ViT) [2].
- Figure 3 shows an overview of the CV4Code transformer models:
  1. **ViT.** Images are split into non-overlapping fixed-size patches. A learnable *[class]* embedding is prepended whose state at the ViT output is the sourcecode representation which is passed to an MLP head.
  2. **ViT-fsd.** While the same setup as ViT is used, we apply shifted patch tokenization and Locality Self-Attention [5].
  3. **Conv-ViT.** To leverage CNN's locality inductive bias we use convolutional layers to create soft visual tokens and keep the use of *[class]* embedding (Figure 3(b)).

Results

Model	Multilingual		Java-only	
	Top-1	Top-5	Top-1	Top-5
ResNet	92.93	96.50	91.17	95.50
ViT-L	92.85	96.86	90.27	95.46
ViT-fsd-L	92.27	96.47	88.99	94.49
Conv-ViT-L	**97.64**	**98.99**	**97.13**	**98.79**

Table 1: *problem_id* classification results on CodeNetBench-Test.

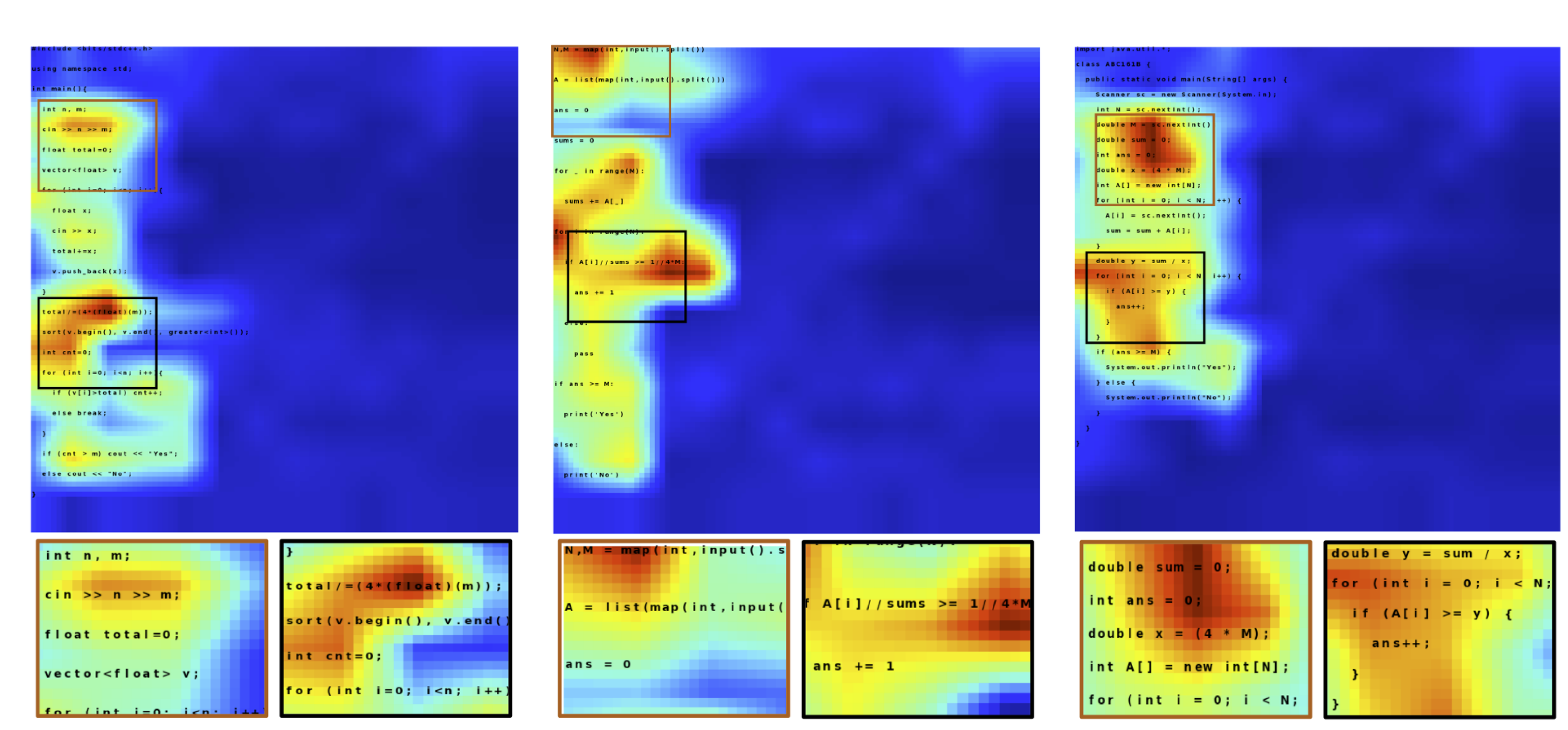


Figure 4: Attention maps (rollout) of Conv-ViT-L. 1) C++, 2) Python, 3) Java

References

[1] Uri Alon, Meital Zilberstein, Omer Levy, and Eran Yahav. Code2vec: Learning distributed representations of code. *Proc. ACM Program. Lang.*, 3(POPL), January 2019.

[2] Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers. *ArXiv*, abs/2104.05704, 2021.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[4] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. 2021.

[5] Seung Hoon Lee, Seunghyun Lee, and Byung Cheol Song. Vision transformer for small-size datasets. *arXiv preprint abs/2112.13492*, 2021.

[6] Rebecca L. Russell, Louis Y. Kim, Lei H. Hamilton, Tomo Lazovich, Jacob A. Harer, Onur Ozdemir, Paul M. Ellingwood, and Marc W. McConley. Automated vulnerability detection in source code using deep representation learning. *CoRR*, abs/1807.04320, 2018.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[8] Daniel Zügner, Tobias Kirschstein, Michele Catasta, Jure Leskovec, and Stephan Günnemann. Language-agnostic representation learning of source code from structure and context. In *International Conference on Learning Representations (ICLR)*, 2021.