EMNLP 2023

Random Entity Quantization for Parameter-Efficient Compositional Knowledge Graph Representation

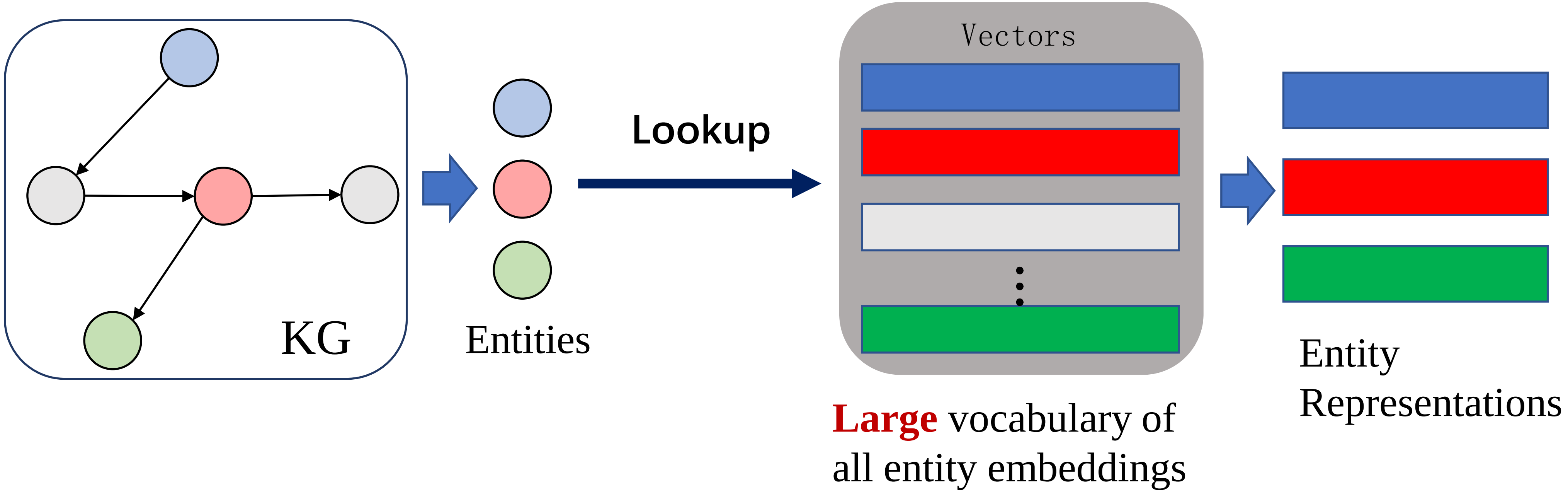
Jiaang Li[1], Quan Wang[2*], Yi Liu[3], Licheng Zhang[1], Zhendong Mao[1]

1: University of Science and Technology of China;
2: MOE Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications;
3: State Key Laboratory of Communication Content Cognition

jali@mail.ustc.edu.cn

Traditional Knowledge Graph Embedding

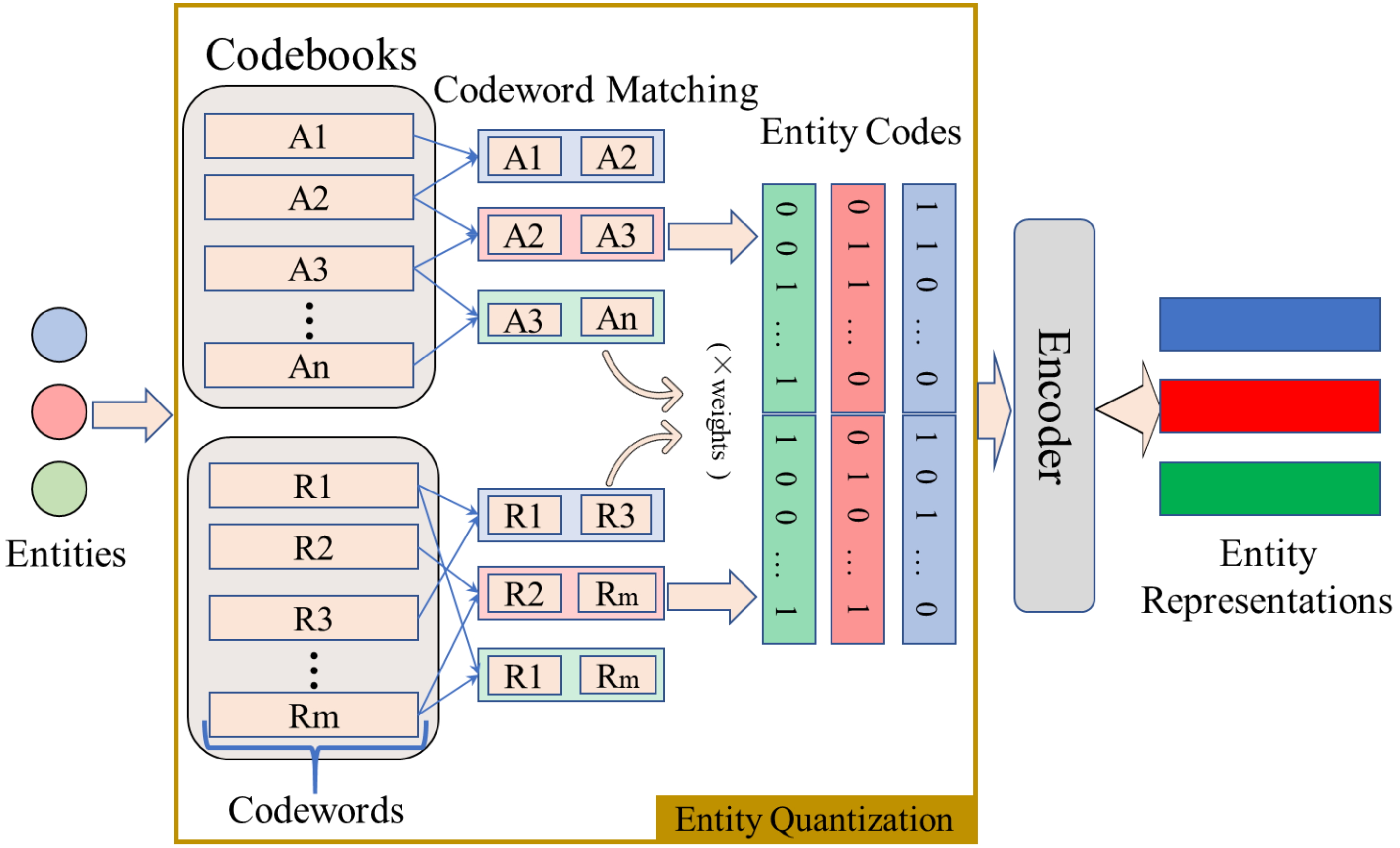
- **Definition: Each entity is represented with an independent vector**



Not scalable! Linear increase in #Parameters with #Entities

Parameter-Efficient Compositional Knowledge Graph Representation

- Pipeline: Represent entities by composing codewords matched by each entity from predefined small-scale codebooks.
- We define this process as Entity Quantization.

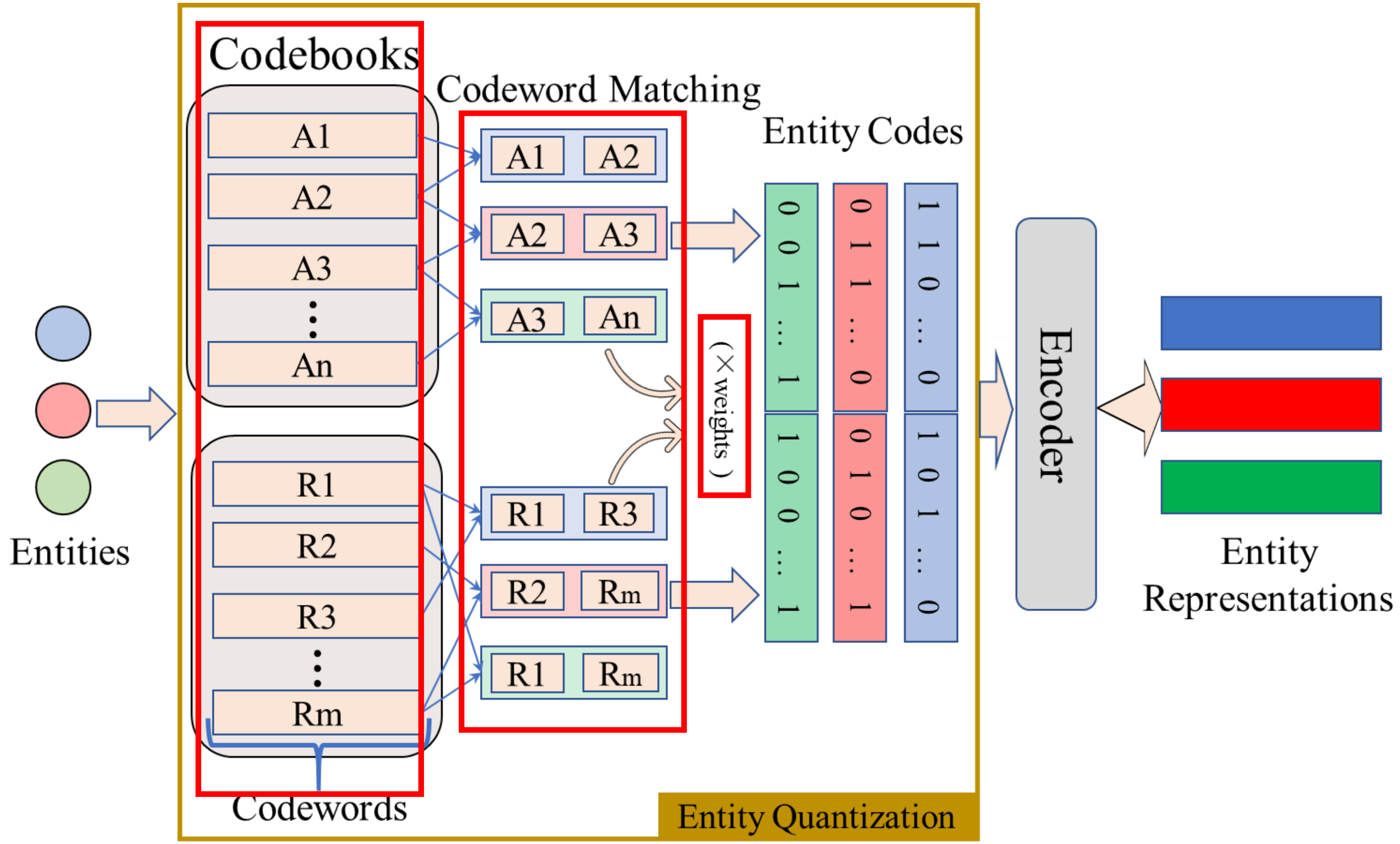


Method	FB15k-237	WN18RR	CoDEx-L
RotatE	29.3 M	40.6 M	78 M
EARL	1.8 M	3.8 M	2.1 M
NodePiece	3.2 M	4.4 M	3.6 M

Reduce lots of params. See Right

Existing Methods

- **Design Complicated strategies based on Connectivity for Entity Quantization.**

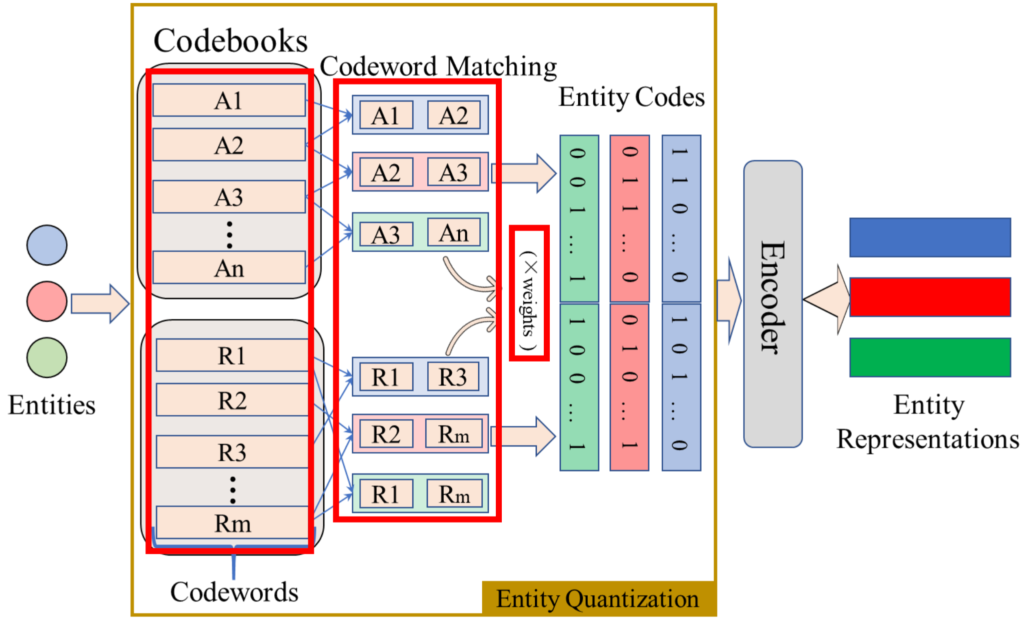


- Codebook construction: 2 codebooks, one with relations, and the other with selected entities (anchors).
- Codeword matching: each entity matches corresponding codewords with rules based on connectivity.
- Codeword weights: Optional. Weights are also assigned with rules based on connectivity.

Are these strategies indispensable?

Our observation: Overview

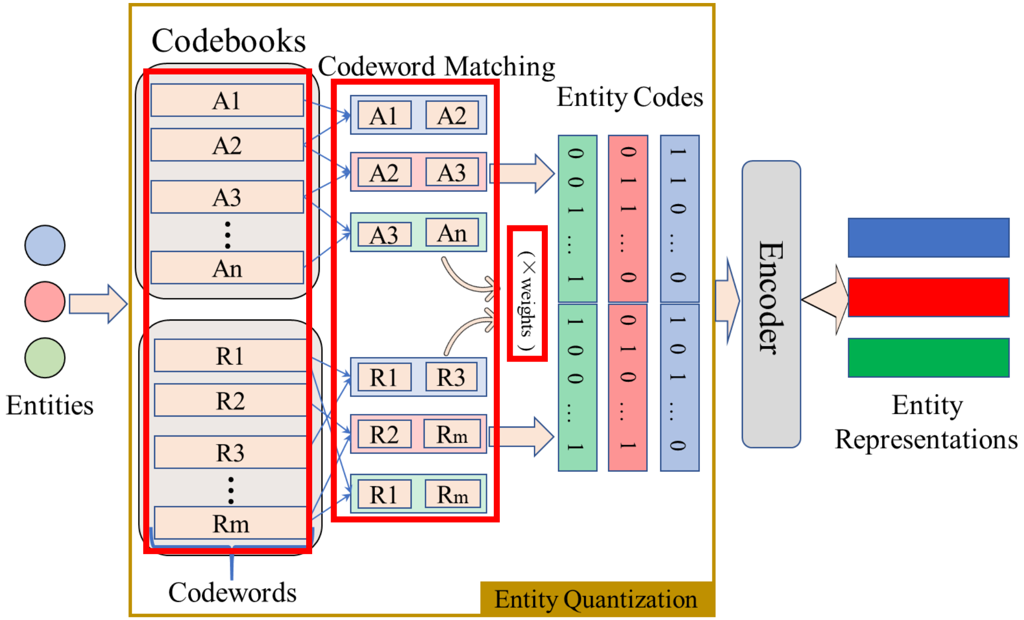
- Randomly quantizing entities won’t affect the overall performance.
- We analyze and explain this phenomenon with entity distinguishability.



We prove previous strategies are not indispensable and analyze why.

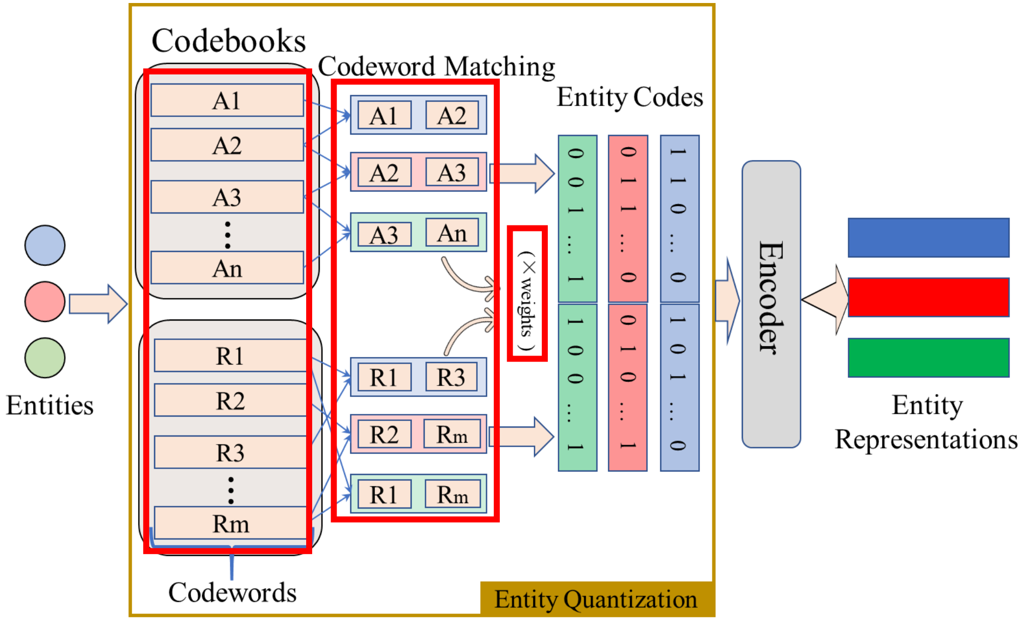
Experiments

- **To prove Random entity quantization won't affect overall performance. Experiments are conducted by random:**
  - codeword matching,
  - codeword weights,
  - codebook construction.



Experiments

- We design random variants by random:
  - codeword matching,
  - codeword weights,
  - codebook construction.



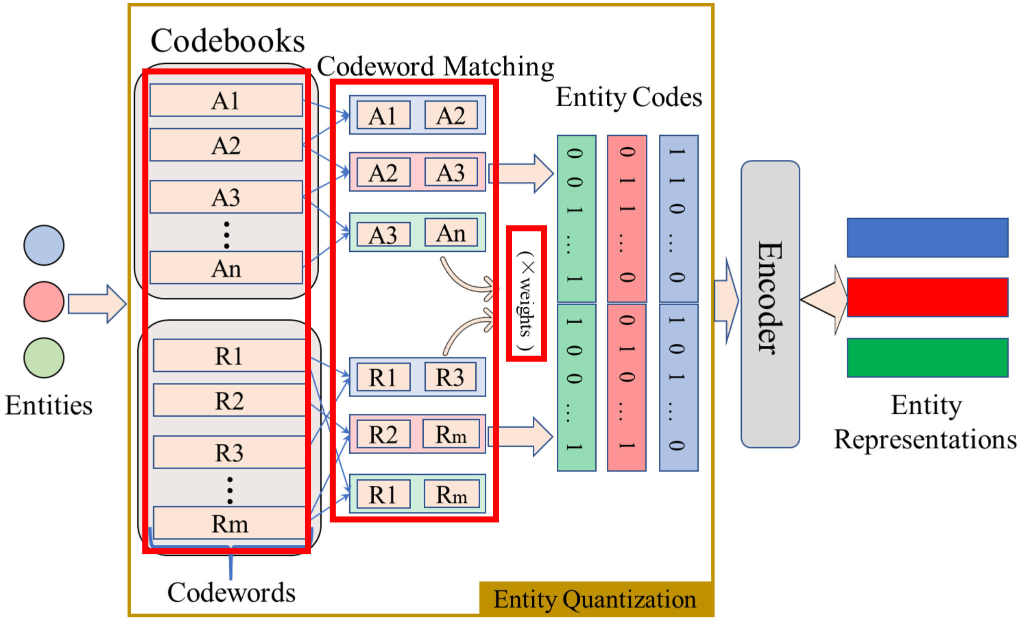
	FB15k-237		WN18RR	
	MRR	Hits@10	MRR	Hits@10
EARL	0.310	0.501	0.440	0.527
+RSR	0.306	0.500	0.439	0.530
+RSA	0.311	0.506	0.438	0.529
+RSR+RSA	0.308	0.502	0.442	0.536
NodePiece	0.256	0.420	0.403	0.515
+RSR	0.254	0.417	0.403	0.516
+RSA	0.258	0.423	0.419	0.518
+RSR+RSA	0.263	0.425	0.425	0.522

	FB15k-237		WN18RR	
	MRR	Hits@10	MRR	Hits@10
EARL	0.310	0.501	0.440	0.527
w/o anc	0.301	0.488	0.409	0.498
w/o anc+RSR	0.312	0.501	0.417	0.516
w/o rel	0.309	0.501	0.432	0.520
w/o rel+RSA	0.311	0.500	0.443	0.539
NodePiece	0.256	0.420	0.403	0.515
w/o anc	0.204	0.355	0.011	0.019
w/o anc+RSR	0.244	0.409	0.009	0.014
w/o rel	0.258	0.425	0.266	0.465
w/o rel+RSA	0.256	0.428	0.411	0.517

Random Entity Quantization is proven to be effective

Experiments

- **We design random variants by random:**
  - codeword matching,
  - codeword weights,
  - codebook construction.

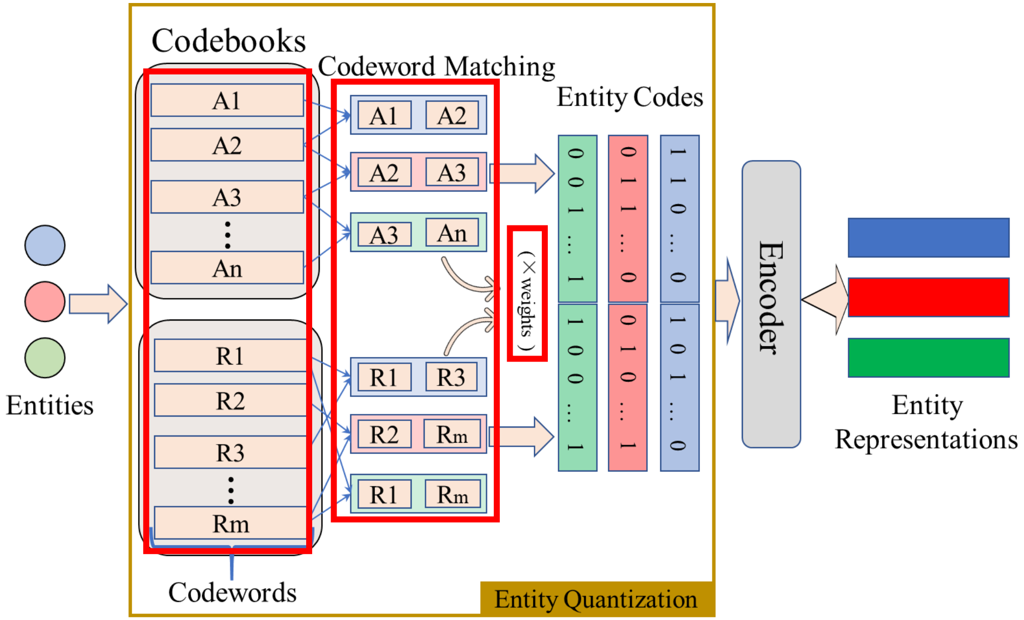


	FB15k-237		WN18RR	
	MRR	Hits@10	MRR	Hits@10
EARL	0.310	0.501	0.440	0.527
+RW	0.308	0.498	0.442	0.531
+EW	0.308	0.500	0.437	0.528

Random Entity Quantization is proven to be effective

Experiments

- **We design random variants by random:**
  - codeword matching,
  - codeword weights,
  - codebook construction.



	FB15k-237				WN18RR				CoDEx-L			
	#P(M)	MRR	Hits@10	Effi	#P(M)	MRR	Hits@10	Effi	#P(M)	MRR	Hits@10	Effi
EARL	1.8	0.310	0.501	0.172	3.8	0.440	0.527	0.116	2.1	0.238	0.390	0.113
EARL+RQ	1.4	**0.311**	**0.505**	**0.222**	3.3	**0.444**	**0.535**	**0.135**	1.9	**0.239**	**0.394**	**0.126**
NodePiece	3.2	0.256	0.420	0.080	4.4	0.403	0.515	0.092	3.6	0.190	0.313	**0.053**
NodePiece+RQ	3.2	**0.261**	**0.423**	**0.082**	4.4	**0.429**	**0.517**	**0.098**	3.6	**0.192**	**0.326**	**0.053**

Random Entity Quantization is proven to be effective

Analysis: Why Random Entity Quantization Works?

- We compute the **entropy of entity codes**
- **Entity codes** have **higher entropy** with **random quantization**

$$H(X) = - \sum_{i=1,\ldots,2^l} P(x_i) \cdot \log_2 P(x_i),$$

$$P(x_i) = \begin{cases} \frac{f_i}{|\mathcal{E}|} & \text{if} \quad i = 1, \ldots, v, \\ 0 & \text{if} \quad i = v+1, \ldots, 2^l. \end{cases}$$

	NodePiece	EARL	Random
FB15k-237	15.26	14.50	15.27
WN18RR	15.94	8.20	16.75

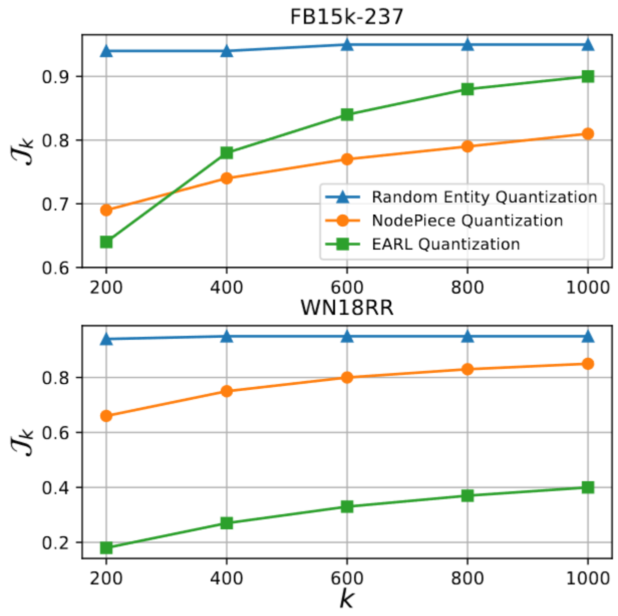
Better distinguishability at the code level!

Analysis: Why Random Entity Quantization Works?

- We compute the **Jaccard distance** between each **entity code** and **its k-nearest neighbors**
- Entities are more **distinct** by **codewords** with **random quantization**

$$d_J(\mathbf{c}_i, \mathbf{c}_j) = \frac{|W_i \cup W_j| - |W_i \cap W_j|}{|W_i \cup W_j|},$$

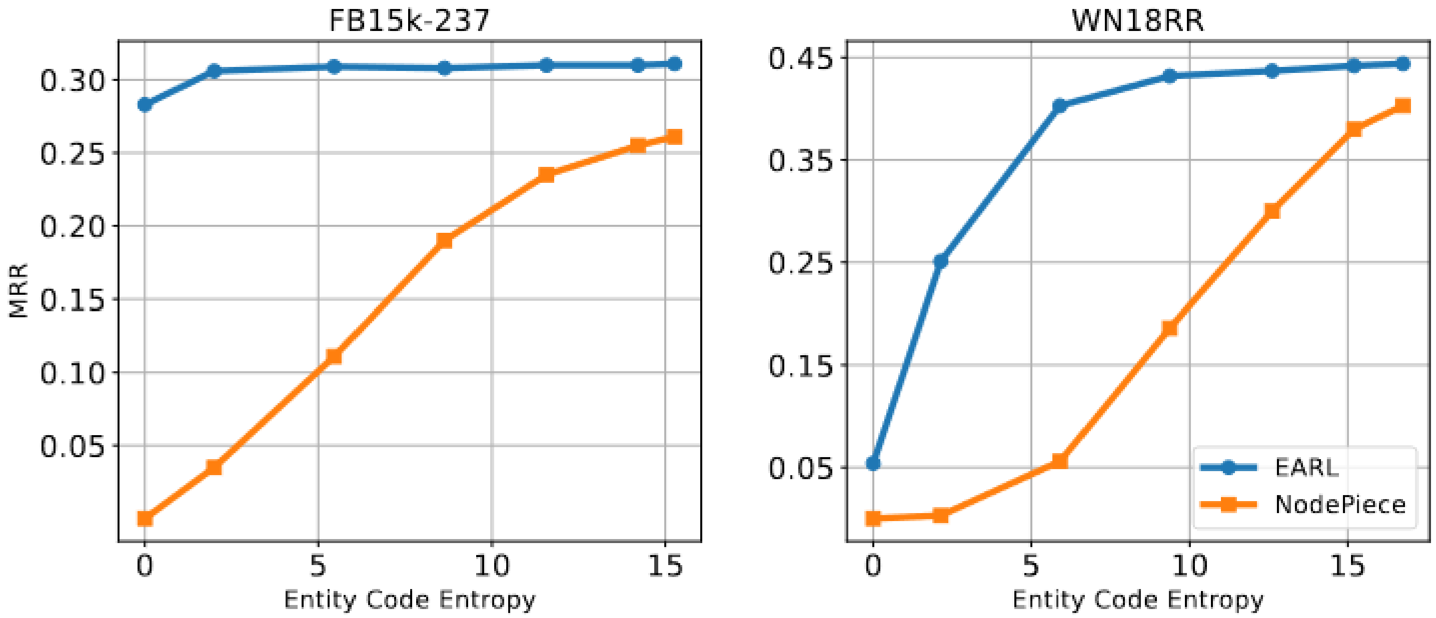
$$\mathcal{J}_k = \frac{1}{|\mathcal{E}| \times k} \sum_{e_i \in \mathcal{E}} \sum_{e_j \in kNN(e_i)} d_J(\mathbf{c}_i, \mathbf{c}_j)$$



Better distinguishability at the codeword level!

Analysis: Why Random Entity Quantization Works?

- Quantization strategies with **higher entity distinguishability** tend to **perform better**



Control the entity code entropy and record results

Entity distinguishability** is why the **random approach** works **comparable to or even better than previous quantization strategies.

Conclusion

- ✓ The first work that defines entity quantization in parameter-efficient compositional KG representation;
- ✓ A new approach based on randomness is proposed for quantizing entities effectively and efficiently;
- ✓ Analyses are made to explain why random entity quantization works comparatively or even better;

Random Entity Quantization for Parameter-Efficient Compositional KGR

Thank you！

Code to reproduce our results is available at: https://github.com/JiaangL/RandomQuantization