A. Proof of Proposition 1

Proof. We first re-parameterize both the codebook $\mathbf{C}$ and the Value matrix $\mathbf{V}$ as follows.

The original codebook is $\mathbf{C} \in \{1, \cdots, K\}^{n\times D}$, and we turn each code bit, which is an integer in $\{1, \cdots, K\}$, into a small one-hot vector of length-K. This results in the new binary codebook $\mathbf{B} \in \{0, 1\}^{n\times KD}$. Per our constraint in proposition 1, $\mathbf{B}$ is a full rank matrix.

The original Value matrix is $\mathbf{V} \in \mathbb{R}^{K\times d}$, and we turn it into a block-diagonal matrix $\mathbf{U} \in \mathbb{R}^{KD\times d}$ where the j-th block-diagonal is set to $\mathbf{V}^{(j)} \in \mathbb{R}^{K\times(d/D)}$. Given that each block diagonal, i.e. $\mathbf{V}^{(j)}$, is full rank, the resulting block diagonal matrix $\mathbf{U}$ is also full rank.

With the above re-parameterization, we can write the output embedding matrix $\mathbf{H} = \mathbf{B}\mathbf{U}$. Given both $\mathbf{B}$ and $\mathbf{U}$ are full rank and $KD \geq d$, the resulting embedding matrix $\mathbf{H}$ is also full rank. □

B. Details of Model Training

We follow the training settings of the base models used, and most of the time, just tune the DPQ hyper-parmeters such as K, D and/or subspace-sharing.

Transformer on WMT'19 En-De. For training the Transformer Model on WMT'19 En-De dataset, the training set contains approximately 27M parallel sentences. We generated a vocabulary of 32k sub-words from the training data using the SentencePiece tokenizer (Kudo & Richardson, 2018). The architecture is the Transformer Base configuration described in (Vaswani et al., 2017) with a context window size of 256 tokens. All models were trained with a batch size of 2048 sentences for 250k steps, and with the SM3 optimizer (Anil et al., 2019) with momentum 0.9 and a quadratic learning rate warm-up schedule with 10k warm-up steps. We searched the learning rate in $\{0.1, 0.3\}$.

BERT pre-training. As our baseline, we pre-train BERT-base (Devlin et al., 2018) on 512-token sequences for 1M iterations with batch size 1024. We used the same optimizer (Adam) and learning rate schedule as described in (Devlin et al., 2018). For the DPQ experiments, we used DPQ-SX with no subspace-sharing, $D = 128$ and $K = 32$, and exactly the same configurations and hyperparameters as in our baseline.

C. Code Study

C.1. Code Distribution

DPQ discretizes the embedding space into the KD codebook in $\{1, ..., K\}^{n\times D}$. We examine the code distribution by computing the number of times each discrete code in each of the D groups is used in the entire codebook:

$$\text{Count}_k^{(j)} = \sum_{i=1}^{n} (\mathbf{C}_i^{(j)} == k)$$

Figure 5 shows the code distribution heat-maps for the Transformer model on WMT'19 En-De, with $K = 32$ and $D = 32$ and no subspace-sharing. We find that 1) DPQ-VQ has a more evenly distributed code utilization, 2) DPQ-SX has a more concentrated and sparse code distribution: in each group, only a few discrete codes are used, and some codes are not used in the codebook.

C.2. Rate of Code Changes

We investigate how the codebook changes during training by computing the percentage of code bits in the KD codebook $\mathbf{C}$ changed since the last saved checkpoint. An example is plotted in Figure 6 for the Transformer on WMT'19 En-De task, with $D = 128$ and various K values. Checkpoints were saved every 600 iterations. Interestingly, for DPQ-SX, code convergence remains about the same for different K values; while for DPQ-VQ, the codes takes longer to stabilize for larger K values.

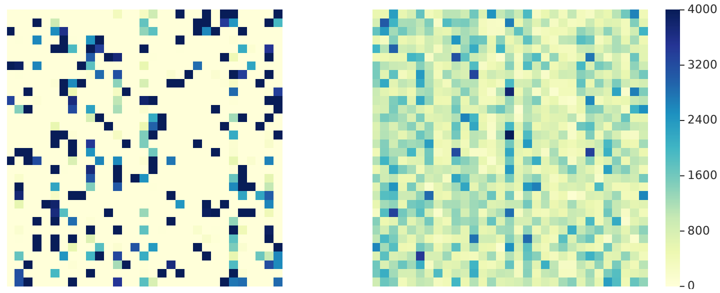

Figure 5. Code heat-maps. Left: DPQ-SX. Right: DPQ-VQ. x-axis: K codes per group. y-axis: D groups. $K = D = 32$.

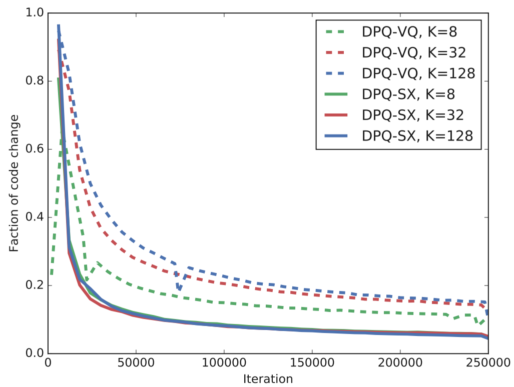

Figure 6. Percentage of code bits in codebook which changed from the previous checkpoint. Transformer on WMT'19 En-De. $D = 128$ for all runs. Checkpoints are saved every 600 iterations.

C.3. Nearest Neighbours of Reconstructed Embeddings

Table 9, 10 and 11 show examples of nearest neighbours in the reconstructed continuous embedding space, trained in the Transformer model on the WMT'19 En-De task. Distance between two sub-words is measured by the cosine similarity of their embedding vectors. Baseline is the original full embeddings model. DPQ variants were trained with $K = D = 128$ with no subspace-sharing.

Taking the sub-word '_evolve' as an example, DPQ variants give very similar top 10 nearest neighbours as the original full embedding: both have 7 out of 10 overlapping top neighbours as the baseline model. However, in DPQ-SX the neighbours have closer distances than the baseline, hence a tighter cluster; while in DPQ-VQ the neighbours are further from the original word. We observe similar patterns in the other two examples.

C.4. Code Visualization

Table 12 shows some examples of compressed codes for both DPQ-SX and DPQ-VQ. Semantically related words share common codes in more dimensions than unrelated words.

Table 9. Nearest neighbours of '_evolve' in the embedding space.

Baseline (Full)	Dist	DPQ-SX	Dist	DPQ-VQ	Dist
_evolve	1.000	_evolve	1.000	_evolve	1.000
_evolved	0.533	_evolved	0.571	_evolved	0.506
_evolving	0.493	_evolution	0.499	_develop	0.417
_develop	0.434	_develop	0.435	_evolving	0.359
_evolution	0.397	_evolving	0.418	_developed	0.320
_developed	0.379	_arise	0.405	_development	0.307
_developing	0.316	_developed	0.405	_developing	0.299
_arise	0.298	_resulted	0.394	_evolution	0.282
_unfold	0.294	_originate	0.361	_changed	0.278
_emerge	0.290	_result	0.359	_grew	0.273

Table 10. Nearest neighbours of '_monopoly' in the embedding space.

Baseline	Dist	DPQ-SX	Dist	DPQ-VQ	Dist
_monopoly	1.000	_monopoly	1.000	_monopoly	1.000
_monopolies	0.613	_monopolies	0.762	_monopolies	0.509
monopol	0.552	monopol	0.714	monopol	0.483
_Monopol	0.380	_Monopol	0.531	_Monopol	0.341
_moratorium	0.271	_zugestimmt	0.486	_dominant	0.258
_privileged	0.269	legitim	0.420	_moratorium	0.239
_unilateral	0.262	_Großunternehmen	0.401	_autonomy	0.230
_miracle	0.260	_Eigenkapital	0.400	_zugelassen	0.227
_privilege	0.254	_wirkungsvoll	0.399	_imperial	0.226
_dominant	0.250	_UCLAF	0.388	_capitalist	0.223

Table 11. Nearest neighbours of '_Toronto' in the embedding space.

Baseline	Dist	DPQ-SX	Dist	DPQ-VQ	Dist
_Toronto	1.000	_Toronto	1.000	_Toronto	1.000
_Vancouver	0.390	_Chicago	0.475	_Orlando	0.307
_Tokyo	0.378	_Orleans	0.467	_Detroit	0.306
_Ottawa	0.372	_Melbourne	0.435	_Canada	0.280
_Philadelphia	0.353	_Miami	0.434	_London	0.280
_Orlando	0.345	_Vancouver	0.415	_Glasgow	0.276
_Chicago	0.340	_Tokyo	0.407	_Montreal	0.272
_Canada	0.330	_Ottawa	0.405	_Vancouver	0.271
_Seoul	0.329	_Azeroth	0.403	_Philadelphia	0.267
_Boston	0.325	_Antonio	0.400	_Hamilton	0.264

Table 12. Examples of KD codes.

	DPQ-SX								DPQ-VQ							
_Monday	2	5	0	7	0	6	1	6	6	5	0	2	4	3	1	7
_Tuesday	6	0	0	7	0	6	1	7	1	7	0	2	0	3	1	7
_Wednesday	6	5	0	3	0	6	1	6	6	2	3	2	0	2	1	7
_Thursday	5	5	0	3	0	6	1	7	7	2	0	2	0	3	1	2
_Friday	4	6	0	7	0	6	1	7	6	0	0	2	1	6	1	7
_Saturday	4	0	6	7	0	6	1	0	6	2	0	2	3	3	1	7
_Sunday	2	0	0	3	0	6	1	6	7	2	0	2	6	3	1	7
_Obama	2	6	7	2	5	7	3	7	2	3	1	6	6	1	7	4
_Clinton	2	4	7	2	3	5	6	7	5	3	5	6	6	0	7	4
_Merkel	4	1	7	2	6	2	2	6	6	3	1	1	4	6	7	4
_Sarkozy	7	6	7	1	4	2	5	0	0	3	1	7	5	7	7	4
_Berlusconi	4	6	5	1	4	2	6	7	6	3	0	6	6	7	7	4
_Putin	2	6	7	1	6	7	6	7	5	3	1	6	6	7	7	6
_Trump	7	6	7	2	0	7	6	7	2	3	1	6	5	7	7	7
_Toronto	6	2	3	2	4	2	2	6	4	3	4	7	6	2	0	7
_Vancouver	2	1	3	2	6	2	5	6	7	3	6	6	6	2	3	1
_Ottawa	2	5	6	1	6	2	2	7	6	3	1	6	6	2	0	4
_Montreal	4	0	0	2	6	2	1	7	4	3	1	1	6	2	0	1
_London	1	2	0	2	4	7	1	7	2	3	0	2	6	3	3	7
_Paris	4	0	3	5	4	2	1	0	5	3	0	0	6	3	2	7
_Munich	4	2	0	4	0	7	5	0	1	3	3	5	6	3	1	7