Differentiable K-Means (DKM)

Soft Clustering for Neural Network Compression

Based on Cho, Vahid, Adya and Rastegari, Apple

ICLR 2022, arXiv:2108.12659v4

November 6, 2025

Outline

Motivation

- Modern deep networks contain millions of parameters
- Model compression reduces storage, bandwidth, and latency
- Clustering-based compression replaces weights with shared centroid
- Challenge: traditional K-Means is non-differentiable
- Approach taken:
  - original loss function and architecture fixed
  - introduce new layer with no learnable parameters
  - based on attention mechanism to capture cluster interactions
  - weight-based clustering
  - efficient multi-dimensional k-means clustering
  - more than 10x compression factor (parameters)
  - appropriate for both computer vision and language modelling tasks

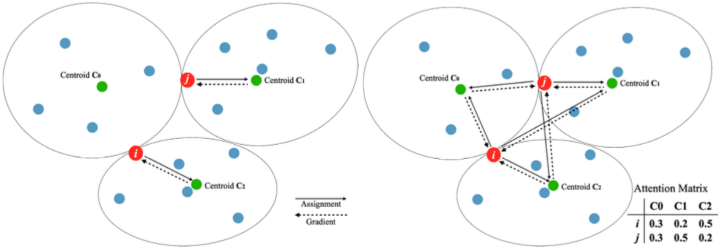
Motivation

- Modern deep networks contain millions of parameters
- Model compression reduces storage, bandwidth, and latency
- Clustering-based compression replaces weights with shared centroid
- Challenge: traditional K-Means is non-differentiable
- Approach taken:
  - original loss function and architecture fixed
  - introduce new layer with no learnable parameters
  - based on attention mechanism to capture cluster interactions
  - weight-based clustering
  - efficient multi-dimensional k-means clustering
  - more than 10x compression factor (parameters)
  - appropriate for both computer vision and language modelling tasks

Goal of DKM

Make the clustering process differentiable and trainable end-to-end with the neural network.

Overcoming limitations



(a) Conventional weight-clustering (Han et al., 2016; Wang et al., 2019b; Stock et al., 2020; J. Lee, 2021)

(b) Attention-based weight-clustering in DKM

- Conventional approaches make hard assignments for i and j based on distance metric
- Gradient is computed only based on the assigned centroid
- Opportunity lost especially with small number of centroids
- Assigning i to C_0 and j to C_2 could be better for training loss given small distance difference
- Centroid weight = distance-based attention optimization
- Gradient of a centroid becomes a product of attentions

Classical K-Means Recap

- N points, K clusters
- Objective - minimize **inertia**:

$$L_{\text{k-means}} = \sum_{i=1}^{N} \min_k \|x_i - \mu_k\|^2$$

- Update rules (Lloyd's algorithm)
- Cluster centroids computed based on the r_{ik} indicator function

$$\mu_k = \frac{\sum_i r_{ik} x_i}{\sum_i r_{ik}}$$

- Non-differentiable because of indicator function:

$$r_{ik} = \begin{cases} 1 & \text{if } k = \arg\min_j \|x_i - \mu_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

Soft Assignments in DKM

- Replace hard assignments by soft, differentiable probabilities
- a_{ik} is the **attention** for sample i w.r.t. centroid k

$$a_{ik} = \frac{\exp(-\frac{\|x_i - \mu_k\|^2)}{\tau})}{\sum_j \exp(-\frac{\|x_i - \mu_j\|^2}{\tau})}$$

  where $\alpha = \frac{1}{\tau}$ is the **sharpness (inverse temperature)**.
- Reconstructed weight:

$$\hat{x}_i = \sum_{k=1}^{K} a_{ik} \, \mu_k$$

Soft Assignments in DKM

- Replace hard assignments by soft, differentiable probabilities
- a_{ik} is the **attention** for sample i w.r.t. centroid k

$$a_{ik} = \frac{\exp(-\frac{\|x_i - \mu_k\|^2)}{\tau})}{\sum_j \exp(-\frac{\|x_i - \mu_j\|^2}{\tau})}$$

where $\alpha = \frac{1}{\tau}$ is the **sharpness (inverse temperature)**.

- Reconstructed weight:

$$\hat{x}_i = \sum_{k=1}^{K} a_{ik}\, \mu_k$$

Temperature Role

Define $T = 1/\alpha$.
High $T \to$ soft assignments; Low $T \to$ near-hard clustering.

Temperature Dynamics

- Attention a_{ik}:

$$a_{ik} = \frac{\exp(-\frac{\|x_i - \mu_k\|^2}{\tau})}{\sum_j \exp(-\frac{\|x_i - \mu_j\|^2}{\tau})}$$

- α (inverse temperature) controls clustering sharpness:

$$\lim_{\alpha \to 0} a_{ik} = \frac{1}{K}, \qquad \lim_{\alpha \to \infty} a_{ik} = \delta_{k, \arg\min_j \|x_i - \mu_j\|^2}$$

- Smooth interpolation between soft and hard assignments.

Temperature Dynamics

- Attention a_{ik}:

$$a_{ik} = \frac{\exp(-\frac{\|x_i-\mu_k\|^2)}{\tau})}{\sum_j \exp(-\frac{\|x_i-\mu_j\|^2}{\tau})}$$

- α (inverse temperature) controls clustering sharpness:

$$\lim_{\alpha\to 0} a_{ik} = \frac{1}{K}, \qquad \lim_{\alpha\to\infty} a_{ik} = \delta_{k,\arg\min_j \|x_i-\mu_j\|^2}$$

- Smooth interpolation between soft and hard assignments.

Practical Schedule

$$\alpha_t = \alpha_0 + (\alpha_{\max} - \alpha_0)\frac{t}{T_{\text{train}}}$$

Gradually increase α for stable training $\rightarrow$ sharp clustering.

Figure: Temperature Effect

- Low α: smooth assignments; High α: peaked responses
- Equivalent to annealing in Gumbel-Softmax relaxations
- Following plot of soft assignment probabilities for different α

Gumbel Soft-Max distribution

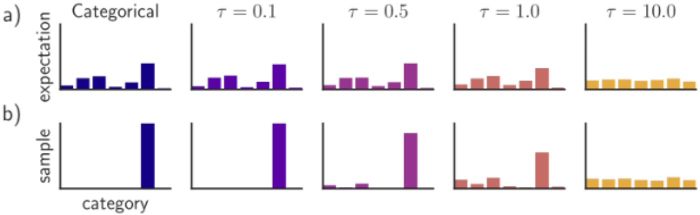
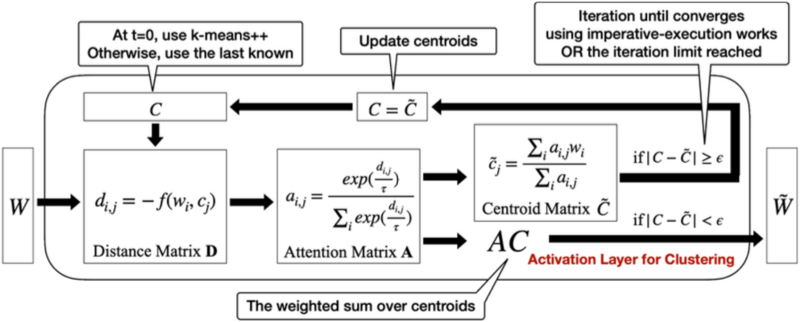


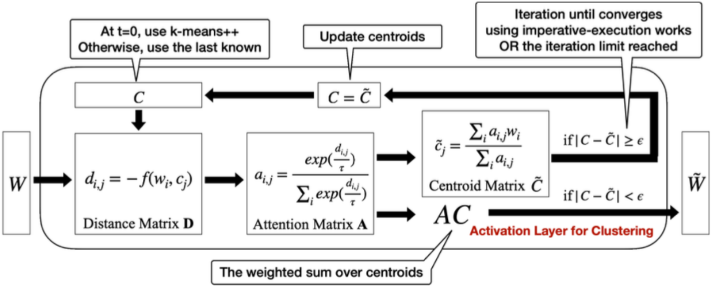
Figure 1: The Gumbel-Softmax distribution interpolates between discrete one-hot-encoded categorical distributions and continuous categorical densities. (a) For low temperatures ($\tau = 0.1, \tau = 0.5$), the expected value of a Gumbel-Softmax random variable approaches the expected value of a categorical random variable with the same logits. As the temperature increases ($\tau = 1.0, \tau = 10.0$), the expected value converges to a uniform distribution over the categories. (b) Samples from Gumbel-Softmax distributions are identical to samples from a categorical distribution as $\tau \to 0$. At higher temperatures, Gumbel-Softmax samples are no longer one-hot, and become uniform as $\tau \to \infty$.

Jang et al, "Categorigal Reparametrization with Gumbel-SoftMax", ICLR 2017, `https://arxiv.org/pdf/1611.01144`

Attention-Like Soft Assignment (1)



- $C \in \mathbb{R}^K$ is the vector of cluster centers and $W \in \mathbb{R}^N$ vector of weights
- initialization of C by randomly select K weights from W or by K-Means++
- distance matrix D computed using differentiable metric (Euclidean distance) f, as $d_{ij} = -f(w_i, c_j)$



- softmax with temperature τ applied on each row of D to get the attention matrix A, a_{ij} from w_i to c_j
- compute centroid candidate $\tilde{C}$ (expectation phase from EM algorihm) gathering contributions into $\tilde{c}_j$
- repeat process until convergence, $|C - \tilde{C}| \leq \epsilon$
- compute $A \cdot C$ to get $\tilde{W}$ for forward-propagation
- since $W = AC$, the iterative process is differentiable for backprop

Compression Ratio

- Replace each weight tensor W by its soft clustered version $\hat{W}$.
- Model size after clustering:

$$S_{\text{compressed}} = K \cdot b_\mu + N \cdot \log_2 K$$

where b_μ is centroid bitwidth.

- Compression ratio:

$$R = \frac{N \cdot b_W}{S_{\text{compressed}}}$$

Result (from Table 3)

ResNet50 $\rightarrow$ 29.4$\times$ compression, -1.6% accuracy drop.

Multi-Dimensional DKM

- N elements of weights are split into $\frac{N}{d}$ contiguous d dimensional sub-vectors
- E.g. flatten all convolutional layers into a $(\frac{N}{d}, d)$ matrix
- Centroids $c_j \in \mathbb{R}^d$ are computed for all subvectors $w_i \in \mathbb{R}^d$
- d-dim subvectors w_i are clustered using b bits per subvector (i.e. 2^b centroids)
- b/d bits for each scalar weight
- Compression Ratio for a 32-bit weight:

$$CR = \frac{32}{b/d}$$

Multi-Dimensional DKM

- N elements of weights are split into $\frac{N}{d}$ contiguous d dimensional sub-vectors
- E.g. flatten all convolutional layers into a $(\frac{N}{d}, d)$ matrix
- Centroids $c_j \in \mathbb{R}^d$ are computed for all subvectors $w_i \in \mathbb{R}^d$
- d-dim subvectors w_i are clustered using b bits per subvector (i.e. 2^b centroids)
- b/d bits for each scalar weight
- Compression Ratio for a 32-bit weight:

$$CR = \frac{32}{b/d}$$

Result

- Keeping same CR we can increase d while adjusting also d increase the entropy of weights' distribution
- Small vectors preserves local structure $\rightarrow$ LUT represents richer patterns

Experimental Results

Model	Compression	Top-1	Drop
ResNet50 (ImageNet)	29.4×	74.5%	-1.6%
MobileNetV1	22.4×	63.9%	-7.0%
DistilBERT	11.8×	83.5%	-1.1%

Observation

- DKM vs. Quantization, Pruning, and Post-hoc Clustering
- Jointly learned centroids outperform discrete quantization
- DKM achieves high compression with minimal loss — outperforming existing clustering methods

Comparison with SotA

		ResNet18	ResNet50	MobileNet-v1	MobileNet-v2
	Base (32 bit)	69.8	76.1	70.9	71.9
3 bit	PROFIT			69.6	69.6
	EWGS	**70.5**	**76.3**	64.4	64.5
	PROFIT+EWGS			68.6	69.5
	DKM	69.9	76.2	**69.9**	**70.3**
2 bit	PROFIT			63.4	61.9
	EWGS	**69.3**	**75.8**	52.0	49.1
	DKM	68.9	75.3	**66.4**	**66.2**
1 bit	PROFIT			nc°	nc
	EWGS	66.6	73.8	8.5	23.0
	DKM 1/1	65.0	72.1	5.9	50.8
	DKM 4/4§	67.0	**73.8**	60.6	55.0
	DKM 8/8	**67.8**	oom□	**64.3**	**62.4**
$\frac{1}{2}$ bit	DKM 4/8	62.1	70.6	46.5	34.0
	DKM 8/16	**65.5**	**72.1**	**59.8**	**58.3**

° not converging; □ out of memory ; § clustering with 4 bits and 4 dimensions

Table 1: When compared with the latest weight quantization algorithms, DKM-based algorithm shows superior Top-1 accuracy when the network is hard to optimize (i.e., MobileNet-v1/v2) or when a low precision is required (1 bit). Further, with multi-dimensional DKM (see Section 3.3), DKM delivers 64.3 % Top-1 accuracy for MobileNet-v1 with the 8/8 configuration which is equivalent to 1 bit-per-weight.

Improved BERT performance

		ALBERT	DistilBERT	BERT-tiny	MobileBERT
	Base (32 bit)	90.6	88.2	78.9	89.6
3 bit	EWGS	83.3	87.6	78.3	87.8
	DKM	**85.1**	**88.2**	**80.0**	**89.0**
2 bit	EWGS	79.6	85.4	77.9	81.6
	DKM	**81.7**	**87.4**	**80.0**	**83.7**
1 bit	EWGS	62.0	60.9	74.5	60.2
	DKM	79.0	82.8	**77.4**	69.8
	DKM 4/4§	**80.0**	**84.0**	77.2	**78.3**

§ clustering with 4 bits and 4 dimensions

Key Takeaways

- DKM introduces a differentiable relaxation of K-Means via temperature-controlled soft assignments.
- Temperature ($T = 1/\alpha$) is crucial:
  - High $T \rightarrow$ stable gradients, fuzzy clustering
  - Low $T \rightarrow$ crisp clustering, high compression
- Proper annealing schedule ensures smooth convergence.
- Memory-heavy to train:

$$O(N + Kd + NK)$$

Last term is the (implicit) attention map, computed during backpropagation

References

- Y. Cho et al., *“DKM: Differentiable K-Means Clustering Layer for Neural Network Compression,”* arXiv:2108.12659, 2021
- S. Lloyd, *“Least squares quantization in PCM,”* IEEE Trans. Inf. Theory, 1982
- E. Jang et al., *“Categorical reparameterization with Gumbel–Softmax,”* ICLR 2017
- S. Jaffe et al. *"IDKM: Memory Efficient Neural Network Quantization via Implicit, Differentiable k-Means"* ICLR 2023