Addressing Token Uniformity in Transformers via Singular Value Transformation (Supplementary Material)

Hanqi Yan[1] **Lin Gui**[1] **Wenjie Li**[2] **Yulan He**[1,3]

[1]Department of Computer Science, University of Warwick, United Kingdom
[2]Department of Computing, The Hong Kong Polytechnic University, China
[3]The Alan Turing Institute, United Kingdom

A PROOF OF THEOREM IN SECTION 3

Theorem: $\forall x \in X^l, \exists x' \in \mathcal{S}^l_{[1,k]}$, where the subspace $\mathcal{S}^l_{[1,k]}$ is defined based on $\lambda_k \geq C \geq \lambda_{k+1}$, then $\|x - x'\|_2 \leq C$.

Proof We assume that X^l can be represented as a $n_l \times m$ matrix:

$$X^l = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_{n_l} \end{bmatrix},$$

where $\vec{x}_i \in \mathbb{R}^m$ is an m-dimensional embedding of a token in the output of l-th layer. After performing SVD on X^l, we have:

$$\begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_{n_l} \end{bmatrix} = \begin{bmatrix} \vec{u}_1 \\ \vec{u}_2 \\ \vdots \\ \vec{u}_m \\ \vdots \\ \vec{u}_{n_l} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_m \end{bmatrix},$$

where the unitary matrix $U = [\vec{u}_1^\top, \vec{u}_2^\top, ..., \vec{u}_{n^l}^\top]^\top$, $V = [\vec{v}_1^\top, \vec{v}_2^\top, ..., \vec{v}_m^\top]^\top$ are $n^l \times n^l$ left singular matrix and $m \times m$ right singular matrix, respectively. Therefore, the two collections of vectors, i.e. $\vec{u}_i = \{u_{i1}, u_{i2}, ..., u_{in^l}\}$ and $\vec{v}_i = \{v_{i1}, v_{i2}, ..., v_{im}\}$, are two subsets of basis for the m-dimensional vector space ($m << n^l$). Without loss of generality, we assume $x_i \in X^l$ can be represented by its corresponding left singular vector, singular values, and the right singular matrix V, which yields:

$$\vec{x}_i = \vec{u}_i \cdot \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_m \end{bmatrix}$$
$$= \begin{bmatrix} \lambda_1 \cdot \vec{u}_1, & \lambda_2 \cdot \vec{u}_2, & \cdots, & \lambda_m \cdot \vec{u}_m \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_m \end{bmatrix} \tag{1}$$

If we separate the singular values into two parts by C, where $\lambda_k \geq C \geq \lambda_{k+1} \geq 0$, we can rewrite Eq. (1) by:

$$\begin{aligned} \vec{x}_i &= \Sigma_{j=1}^m \lambda_j \cdot u_{ij} \cdot \vec{v}_j \\ &= \Sigma_{j=1}^k \lambda_j \cdot u_{ij} \cdot \vec{v}_j + \Sigma_{j=k+1}^m \lambda_j \cdot u_{ij} \cdot \vec{v}_j \end{aligned}$$

By defining $\vec{x}'_i = \Sigma_{j=1}^k \lambda_j \cdot u_{ij} \cdot \vec{v}_j$, where singular values are taken from the larger group, we have:

$$\begin{aligned} ||\vec{x}_i - \vec{x}'_i|| &= ||\Sigma_{j=k+1}^m \lambda_j \cdot u_{ij} \cdot \vec{v}_j|| \\ &= |<\vec{\lambda}^{[k+1,m]} \otimes \vec{u}_i^{[k+1,m]}, V^{(m-k-1)\times m}>| \end{aligned}$$

Where $||\cdot||$ is the norm, $\otimes$ is the pairwise product, and $|<\cdot,\cdot>|$ is the inner product in a vector space, $\vec{\lambda}^{[k+1,m]}$, and $\vec{u}_i^{[k+1,m]}$ are the sub-vectors of singular values and $\vec{u}_i$ from $k+1$-th to m-th dimensions, respectively, and $V^{(m-k-1)\times m}$ is the corresponding right singular sub-matrix. According to Hölder inequality, we have:

$$||\vec{x}_i - \vec{x}'_i|| \leq ||\vec{\lambda}^{[k+1,m]} \otimes \vec{u}^{[k+1,m]}|| \cdot ||V_{(m-k-1)\times m}||$$

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

Since V is a unitary matrix, $V^\top \cdot V = I$, which yields $||V_{(m-k-1)\times m}|| = 1$. Hence,

$$\begin{aligned}||\vec{x}_i - \vec{x}'_i|| &\le ||\vec{\lambda}^{[k+1,m]} \otimes \vec{u}^{[k+1,m]}|| \\ &= \sqrt{\Sigma_{j=k+1}^m \lambda_j^2 \cdot u_{ij}^2}\end{aligned}$$

Considering $||\vec{u}|| = 1$ and $\lambda_{k+1} \le C$, obviously we have $||\vec{u}_{[k+1,m]}|| \le 1$ and $\lambda_j \le C$, when $j \ge k+1$. Therefore,

$$||\vec{x}_i - \vec{x}'_i|| \le C \cdot \sqrt{\Sigma_{j=k+1}^m u_{ij}^2} \le C$$

□

A case study where the vectors in the unitary matrix U follows a uniform distribution in a L_2-norm based metric space

The **theorem** states that the learned features from a transformer-based language model can be represented as a closure which is defined as a C-neighbour of a k-dimensional space. Here, we present a case study, assuming the vectors $\vec{u}_i$ in the unitary matrix U follows a uniform distribution within a L_2-norm based metric space.

Under such an assumption, the probability of $P(\Sigma_{j=k+1}^m \sqrt{u_{ij}^2} \le d)$ is the integral of the probability density function in the corresponding area of a n-sphere, denoted as S_{n-1}, defined by $\Sigma_{j=k+1}^m \sqrt{u_{ij}^2}$. It is clear that $\mathrm{P}(\Sigma_{j=k+1}^m \sqrt{u_{ij}^2} \le d) \ge 0$. Hence, we only discuss the upper boundary of P in the following. We denote the sub-area of $\Sigma_{j=k+1}^m \sqrt{u_{ij}^2} \le d$ as S_ϕ. To simplify the notation, without loss of generality, we re-order the elements in $\vec{u}_i \in \mathbb{R}^n$ such that its last k dimensions correspond to the small singular values. Then, we have

$$\begin{aligned}&\mathrm{P}(\vec{u}_i \in S_\phi) \\ =&\int_{S_\phi} \frac{\Gamma(n/2)}{2\pi^{n/2}} \Pi_{i=1}^{n-2} \sin^{n-1-i}(\psi_i) d\phi_1 ... d\phi_{(n-1)} \\ \le&\frac{\Gamma(n/2)}{2\pi^{n/2}} \cdot d^k \int_{S_\phi} \Pi_{i=1}^{n-k} \sin^{n-k-i}(\psi_i) d\phi_1 ... d\phi_{(n-k+1)} \cdot \\ &\int_{S_\phi} \Pi_{i=1}^{k} \sin^{k-i}(\psi_{n-k-1+i}) d\phi_k ... d\phi_{(n-1)} \\ \le&\frac{\Gamma(n/2)}{2\pi^{n/2}} \cdot \frac{2\pi^{(n-k)/2}}{\Gamma((n-k)/2+1)} \cdot \frac{2\pi^{k/2}}{\Gamma(k/2+1)} \cdot (1-d^{n-k}) \cdot d^{2k} \\ \le&\frac{2}{k(n-k)} \cdot \frac{\Gamma(n/2)}{\Gamma((n-k)/2)\Gamma(k/2)} \cdot d^{2k}(1-d^{n-k}) \\ =&\frac{2}{k(n-k) \cdot \mathrm{B}(k/2,(n-k)/2)} \cdot d^{2k}(1-d^{n-k}), \qquad (2)\end{aligned}$$

where $\mathrm{B}(\cdot,\cdot)$ is the beta function. The result show that the probability of a singular vector residing in the sub-area S_ϕ will converge to 0 exponentially with the growth of k. As such, when k, the number of smaller singular vectors, is large, the distance between the embedding space and the subspace spanned by the larger singular vectors is bounded by C, the smallest value in the larger singular value group.

B MODEL CONFIGURATIONS AND TRAINING DETAILS

Unsupervised Setting In the unsupervised setting on the STS task, we use the datasets processed by [Huang et al., 2021] and follow their evaluation pipeline by replacing their Whitening function with our `SoftDecay` function in their released code[1]. We do not use any dataset to train the transformation function, instead, we choose a fixed α empirically (α is the hyper-parameter in Eq.(3)). As we did not see significant changes across different α, we set α to -0.6 for all the datasets and PTLMs. For metrics calculation, we use $t = 0.5$ in RBF$_{dis}$ and we choose the nearest 12 points to reconstruct the query point in *LSDS*.

Supervised Setting We apply `SoftDecay` to the output of the last layer of a PTLM provide by huggingface, before layer normalisation. We use the default parameters configured in BERT-base-uncased[2], ALBERT-base-v1[3], RoBERTa-base[4] and DistilBERT-base-uncased[5] as the baselines. For hyper-parameter setting, we search the initial alpha for different datasets from $[-0.2, -0.5, -0.8]$, and set different learning rates from $[2e-3, 2e-5]$ for the transformation layer and the pretrained models.[6]

C ADDITIONAL RESULTS ON SEMANTIC TEXTUAL SIMILARITY DATASET

In this section, we first examine the potential reasons of improvement by comparing the learnt representations from baselines models (i.e., vanilla PLTMs and Whitening-BERT) and our proposed `SoftDecay` through quantitative evaluation results and the visualisation results (See in §c.1. and §c.2). We then discuss a comparison between `SoftDecay` and a representative contrastive learning method, `SimCSE` [Gao et al., 2021], which also aims to

[1] `https://github.com/Jun-jie-Huang/WhiteningBERT`

[2] `https://huggingface.co/docs/transformers/master/en/model_doc/bert`

[3] `https://huggingface.co/docs/transformers/master/en/model_doc/albert`

[4] `https://huggingface.co/docs/transformers/master/en/model_doc/roberta`

[5] `https://huggingface.co/docs/transformers/master/en/model_doc/distilbert`

[6] As SVD decomposition generates an error in the RoBERTa-base model, we exclude it in GLUE evaluation.

		BERT	+SoftDecay	ALBERT	+SoftDecay	DistilBERT	+SoftDecay
STS-B	Evs	0.6259	0.0252	0.6987	0.0326	0.7301	0.0341
	RBF$_{dis}$	-1.4624	-3.8534	-1.1602	-3.8016	-1.0549	-3.8052
	TokenUni	0.6195	0.0274	0.6983	0.036	0.7282	0.037
SICK	Evs	0.7383	0.0212	0.7711	0.0274	0.8135	0.0289
	RBF$_{dis}$	-1.0323	-3.8671	-0.8979	-3.8268	-0.7367	-3.8241
	TokenUni	0.7361	0.023	0.7706	0.0295	0.8130	0.0311
STS-12	Evs	0.6219	0.0182	0.7052	0.0247	0.7321	0.0245
	RBF$_{dis}$	-1.4785	-3.8717	-1.4785	-1.1438	-3.8308	-3.8381
	TokenUni	0.6193	0.0203	0.7058	0.0273	0.7021	0.0329
STS-13	Evs	0.5823	0.0221	0.6632	0.0287	0.7015	0.0302
	RBF$_{dis}$	-1.6189	-3.8706	-1.3032	-3.8258	-1.1594	-3.8262
	TokenUni	0.5817	0.024	0.6637	0.031	0.7021	0.0329
STS-14	Evs	0.5933	0.6729	0.0204	0.0151	0.712	0.0202
	RBF$_{dis}$	-1.593	-3.9124	-1.2712	-3.8787	-1.1288	-3.8855
	TokenUni	0.5929	0.016	0.6743	0.0217	0.7127	0.0215
STS-15	Evs	0.6072	0.0183	0.6827	0.0239	0.7225	0.0248
	RBF$_{dis}$	-1.5177	-3.8706	-1.2178	-3.8379	-1.0772	-3.8313
	TokenUni	0.6057	0.0216	0.6848	0.0273	0.7228	0.0291
STS-16	Evs	0.6049	0.0267	0.6824	0.0333	0.7190	0.0363
	RBF$_{dis}$	-1.5262	-3.8375	-1.5262	-1.2095	-3.7952	-3.7869
	TokenUni	0.6054	0.0286	0.6864	0.0360	0.7201	0.0390

Table 1: Uniformity metrics (*EVs*, *TokenUni*, *RBF$_{dis}$*) evaluates the isotropy in transformed feature space comparing to the vanilla PTLMs features. Smaller values means the features are better uniformly distributed. It can be seen that `SoftDecay` can greatly improve the uniformity.

alleviate the anisotropy problem in language representations.

C.1 FEATURE EVALUATION RESULTS ON STS DATASETS

We show in Table 1 and Figure 1 both the uniformity and local neighborhood preservation evaluation results of different methods over the seven STS datasets. The lower scores returned by `SoftDecay` in Table 1 in comparison to the base PTLMs verify its capability of alleviating anisotropic feature space derived from `BERT`. In Figure 1), `SoftDecay` preserves the local neighbourhood structure better among all the datasets, which explains its performance superiority comparing with `Whitening` which ignores the original local manifold structure.

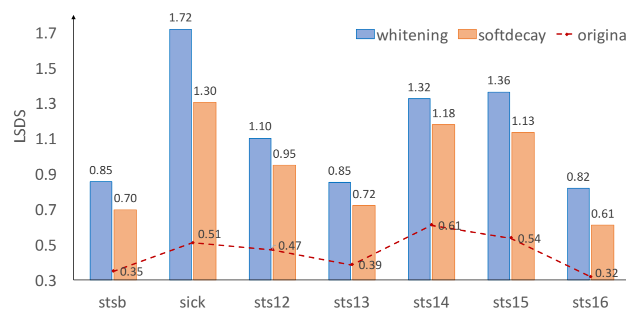

Figure 1: Local Structure Discrepancy Score (*LSDS*) for `Whitening` and `SoftDecay` transformed Representations. Smaller scores are preferred as the original local neighborhood information learnt in the pretrained model is preserved better.

C.2 VISUALISATION OF FEATURES IN STS DATASETS

We show the representations of sentence pairs generated from `BERT`, with `Whitening` and with `SoftDecay` via tSNE for the rest five STS datasets in Figure 2. In STSB, STS13 and STS16, the representation mapping results in `Whitening` are not unit Gaussian due to some *abnormal* data point. Our proposed method `SoftDecay` gives better uniformity score than vanilla BERT and better *LSDS* than `WhiteningBERT`, as have been shown in Figure 1 and Table 1.

C.3 COMPARISON WITH CONTRASTIVE LEARNING ON STS

The objective of contrastive learning methods is to align semantically-related positive data pairs and make the learned representations evenly distributed in the resulting embedding space [Wang and Isola, 2020]. The latter property naturally addresses the token uniformity issue. Therefore, we further compare `Softdecay` with a representative contrastive learning method, `SimCSE` [Gao et al., 2021], on STS. As SimCSE needs to be trained on datasets to fine-tune its parameters, we conduct experiments using `SimCSE` fol-

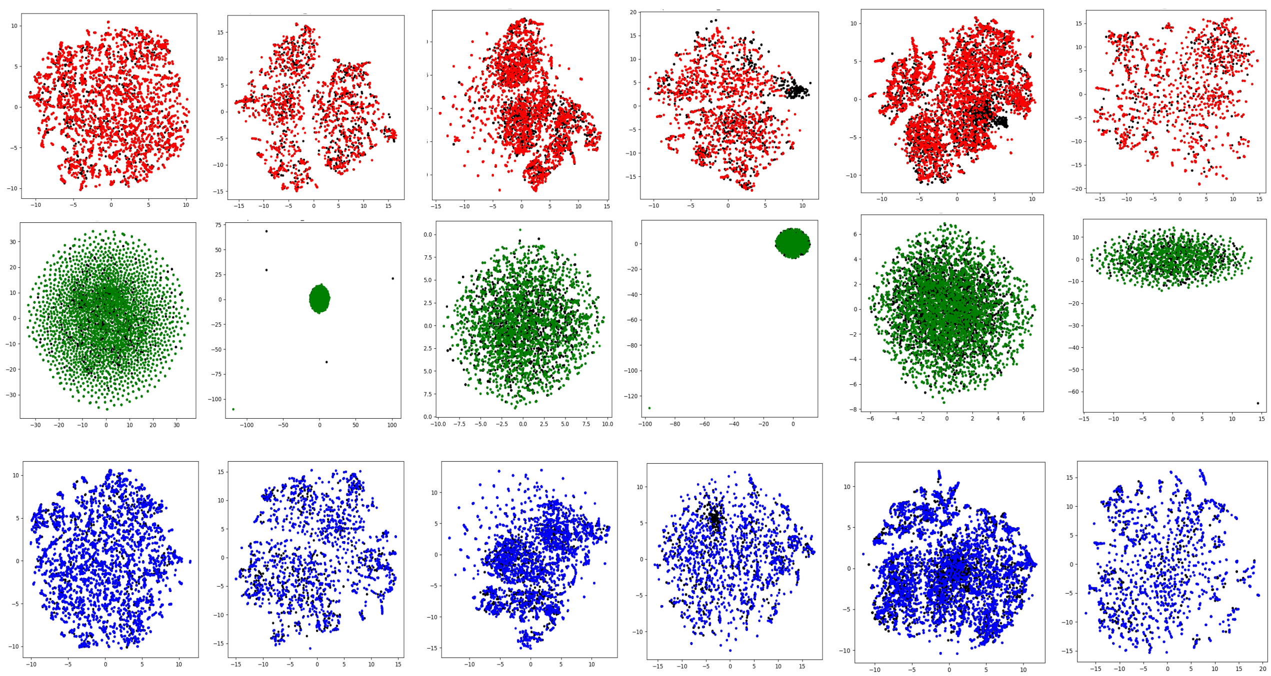

Figure 2: The tSNE visualisation of representations of sentence pairs in datasets SICKR, STSB, STS12-16 (except STS15) in different columns. These representations from top to bottom are derived from vanilla `BERT`, `BERT+whitening` and `BERT+SoftDecay`. For each sentence pair, the two sentences are denotes by different colors, e.g., black and red in `BERT`. We can see clear clusters in `BERT` and `BERT+SoftDecay` for STS-B, STS-12 and STS-14 datasets.

lowing its original setup: (1) *Unsupervised*. Train the model on sampled 1 million sentences from English Wikipedia [7] and pass the same sentence twice to a pre-trained encoder with standard dropout to generate two different sentence embeddings as positive pairs. Other sentences in the same mini-batch are taken as negative pairs; (2) *Supervised*. Train the model on natural language inference datasets, MNLI and SNLI [8], and use the annotated entailment and contradictory pairs as positive and negative sentence pairs, respectively. The results are shown in Table 2. It can be observed that `SoftDecay` outperforms `SimCSE` in general, especially under the supervised setting. The end goal of our approach (via increasing the weights of small singular values in the output embedding space) is similar to `SimCSE` (via random dropout masks) under the unsupervised setting, as both aim to learn an isotropic embedding distribution. However, in the supervised `SimCSE`, its contrastive loss is calculated on a subset of training pairs, as such, it is relatively difficult to achieve the universal isotropy, which is not the case in our approach.

[7]Download link for Sampled English Wikipedia dataset

[8]Download link for the combined NLI dataset

D ADDITIONAL RESULTS ON GLUE DATASETS

In this section, we first show the results of comparing `SoftDecay` with another method, which applies regularisation during training to alleviate the anisotropy issue. Then, we display the Cumulative distribution function (CDF) of singular value distributions before and after applying `SoftDecay`.

D.1 COMPARING WITH ANOTHER SINGULAR VALUE TRANSFORMATION FUNCTION

In addition to Sentence-BERT (`S-BERT` for short) [Reimers and Gurevych, 2019] and `BERT-CT` [Carlsson et al., 2021], we also compare with another method which applies regularisation on the output embedding matrix with an exponentially decayed singular value prior distribution during

Model	STSB	STS-12	STS-13	STS-14	STS-15	STS-16	SICK-R
Trained on wiki-text (unsupervised)							
`SimCSE` [Goyal et al., 2020]	74.48	**66.01**	**81.48**	**71.77**	77.55	76.53	69.36
`SoftDecay`	**75.81**	63.25	78.67	70.41	**79.37**	**77.69**	**71.15**
Trained on MNLI and SNLI dataset (supervised)							
`SimCSE` [Goyal et al., 2020]	82.26	**77.37**	78.12	77.81	84.65	81.10	78.73
`SoftDecay`	**83.51**	75.31	**81.70**	**79.88**	**86.33**	**81.37**	**79.04**

Table 2: Comparison with contrastive learning method, `SimCSE`. Our methods demonstrate overall better results under the supervised setting.

training (`ExpDecay` for short) [Wang et al., 2020].

`ExpDecay` is designed for an encoder-decoder architecture in language generation. The singular value distribution of the output embedding matrix is derived from the decoder. This approach is not directly applicable to our setup since we don't use the encoder-decoder architecture here. Nevertheless, we modify our training objective by adding the singular values $\{\lambda_k\}_{k=1}^K$ of output feature X: $\gamma e \sum_{k=1}^K (\lambda_k - c_1 e^{-c_2 k^\gamma})$. where γe is a hyperparameter used to adjust the weight of the added term, c_1, c_2, and γ are hyperparameters in the desirable exponential prior term of singualar values. We empirically set $c_1, c_2 = 1, \gamma = 2, \gamma e = 1e - 4$.

By comparing with the results of `ExpDecay` in Table 3, we don't see substantial improvement using the fixed exponential decay term. It can be explained by 1) the difficulty of balancing two losses by adding the exponential decay term into the training objective function; 2) the sensitivity of the hyper-parameter in the prior decay term in `ExpDecay`. In our method, we only has a single parameter α in Eq. (2) and its value can be automatically adjusted during training to fit the downstream tasks under the supervised setting.

Dataset (size)	BERT	+`SoftDecay`($\Delta\%$)	+`ExpDecay`($\Delta\%$)
CoLA(8.5k)	59.57	**59.84***(↑0.45)	59.37(↓0.34)
SST2(67k)	92.32	**93.12****(↑0.87)	**92.43**(↑1.19)
MRPC-Acc(3.7k)	84.00	**85.20****(↑1.43)	83.25(↓0.89)
MRPC-F1(3.7k)	89.50	**89.65**(↑0.17)	87.92(↓1.21)
QNLI(105k)	91.25	**91.98****(↑0.80)	89.21(↓2.23)
RTE(2.5k)	64.98	**68.23****(↑5.00)	64.98(↑0.00)

Table 3: Sentence-level classification results on five representative GLUE validation datasets. Matthews correlation is used to evaluate CoLA, Accuracy/F1 is used in other datasets. $\Delta\%$ represents the relative improvement over the baseline. Better results than BERT are in bold. No substantial improvements are observed using `ExpDecay`.

D.2 SINGULAR VALUE DISTRIBUTION

The effects of dataset size on NLI dataset We highlight the different singular value distribution in QNLI and RTE, two datasets for language inference task (See in Figure 3).

BERT-Based Model Results For BERT-based model, we show the CDF of singular values on all the evaluated datasets in Figure 4. We observe that by applying `SoftDecay` (bottom row of Figure 4), the CDF of singular values in the last layer becomes more flattened compared to that in vanilla BERT (top row of Figure 4).

ALBERT-Based and DistilBERT-based Model Results We also show the results for `ALBERT` (Figure 5 and Figure 6) and `DistilBERT` (Figure 7 and Figure 8). By comparing with the vanilla PTLMs (the top row of each figure), we notice that the application of `SoftDecay` has a larger impact on ALBERT compared to DistilBERT, especially on the CoLA dataset. For `DistilBERT`, its feature space becomes anisotropic gradually as layers go deeper.

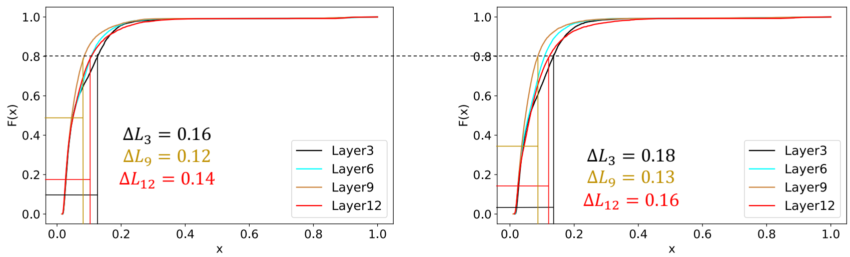

Figure 3: The CDF of singular value in QNLI (left) and RTE (right) dataset derived from vanilla `BERT`. For the same percentage 0.8, the larger dataset QNLI dataset has smaller ΔL_i among all the layers, refers to a more serious token uniformity issue.

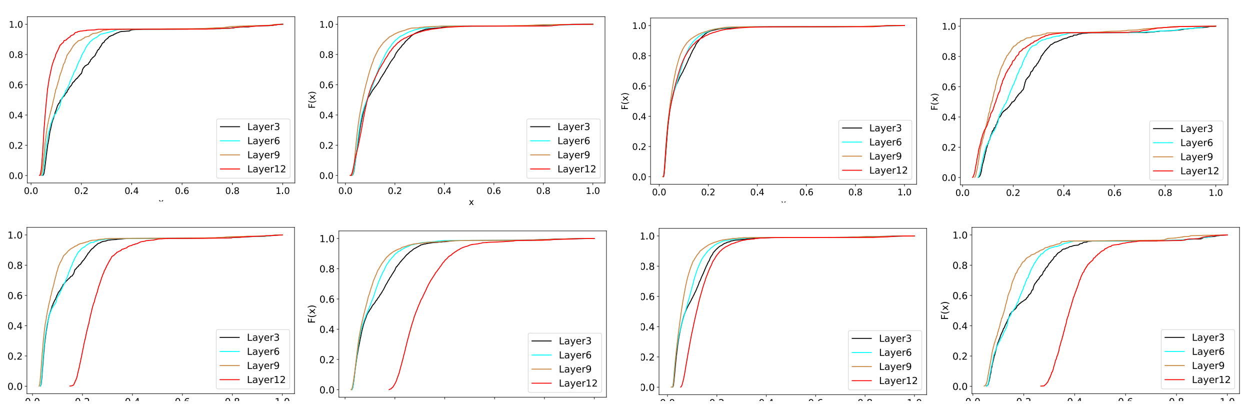

Figure 4: Cumulative distribution function (CDF) of singular value distributions. The upper ones are from vanilla `BERT`, bottom ones are from `BERT+SoftDecay`. From left to right, the evaluation datasets are SST-2, MRPC, QNLI and CoLA. Different curves represent distributions derived from different model layers. The x-axis represents the normalised singular values sorted in an ascending order. `SoftDecay` adjusts the anisotropy of the feature space with the effect more noticeable in MRPC and less obvious in QNLI.

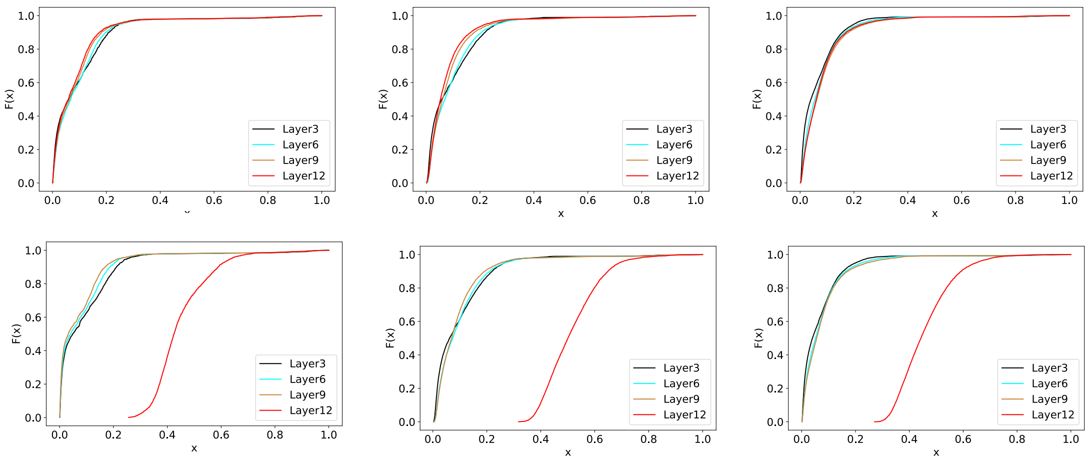

Figure 5: CDF of SST-2, MRPC and QNLI datasets. The upper row results are from the vanilla `ALBERT`, the bottom ones are from `ALBERT+SoftDecay`.

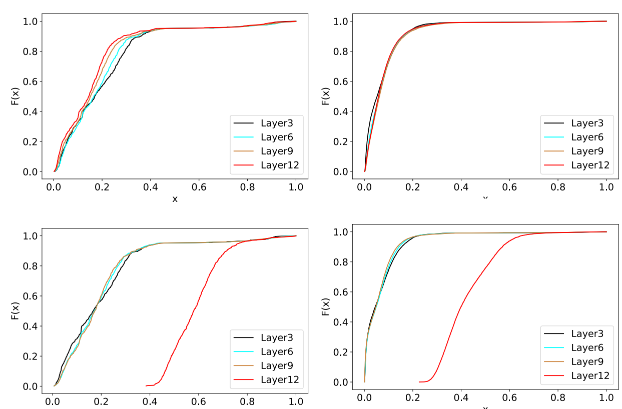

Figure 6: CDF of CoLA and RTE datasets. The upper row results are from the vanilla ALBERT, the bottom ones are from `ALBERT+SoftDecay`.

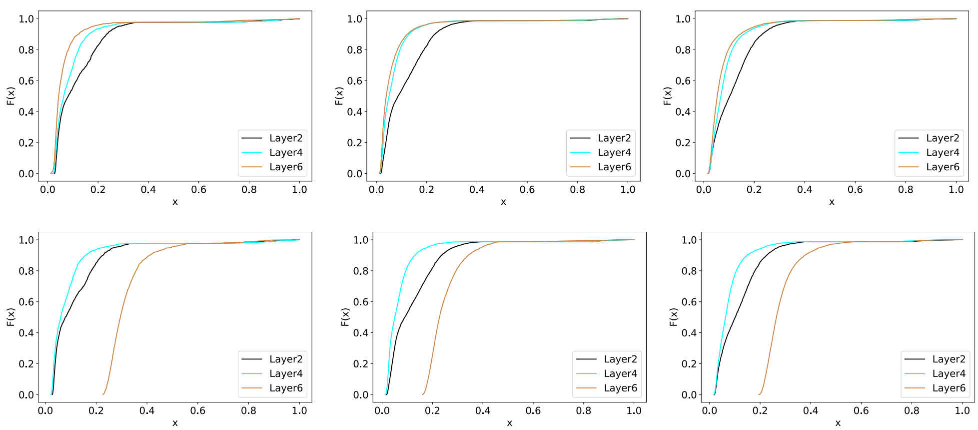

Figure 7: CDF of SST-2, MRPC and QNLI datasets. The upper row results are from the vanilla DistilBERT, the bottom ones are from `DistilBERT+SoftDecay`.

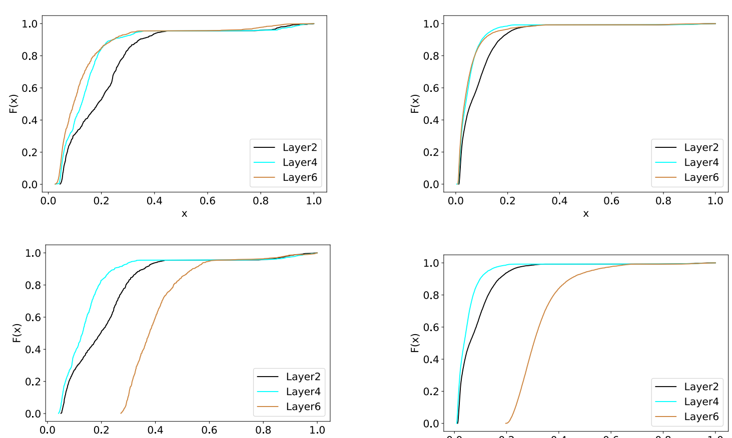

Figure 8: CDF of CoLA and RTE datasets. The upper row results are from the vanilla DistilBERT, the bottom ones are from `DistilBERT+SoftDecay`.

References

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. Semantic re-tuning with contrastive tension. In *9th International Conference on ICLR 2021, Virtual Event, Austria*, 2021.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.552.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th ICML*, volume 119, pages 3690–3699, 2020.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. Whiteningbert: An easy unsupervised sentence embedding approach. *CoRR*, abs/2104.01767, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on EMNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990, 2019. doi: 10.18653/v1/D19-1410.

Lingxiao Wang, Jing Huang, Kevin Huang, Ziniu Hu, Guangtao Wang, and Quanquan Gu. Improving neural language generation with spectrum control. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR, 2020. URL `http://proceedings.mlr.press/v119/wang20k.html`.