Con-
trastive
Weight
Ty-
ing
$_2/headless/imgs/hlm_basic.pdf Masked Headless Language Modeling (HLM) using Contrastive Weight Tying. The CWTo$
$\times$
$\times$
$_1$
$_train_bengio_03, mnih2012fast, jean-$
etal−
2015−
using, ma−
collins−
2018−
noise. These methods approximate the denominator of the softmax by using only a subset of the possible tokens. Those approach
Contrastive Estimation objective ence that use unique negative samples, contrary to our approach that samples representation
similarity to match pre−
trained static embeddings for Machine Translation. We instead use the model's input embeddings as a trainable target represen
$_interspeech, wav2vec2, algayres-$
etal−
2022−
dp. In NLP, contrastive learning has proven efficient in the training of sentence−
level models gao − etal − 2021 − simcse, yan − etal − 2021 − consert, klein − nabi − 2023 − micse. Token−
level approaches rely on contrastive auxiliary objectives that are added to the usual cross−
entropy loss. SimCTG su2022contrastive introduces a token−
level contrastive objective using in−
batch output representations as negative samples, and adds this objective to a sentence−
level contrastive loss and a regular causal LM loss. TaCL su − etal − 2022 − tacl relies on a similar technique for encoder mod
$X =$
$(x_{i,j})_{i \in [1,N], j \in [1,L]}$
N
L
$\tilde{X} =$
$(\tilde{x}_{i,j})_{i \in [1,N], j \in [1,\tilde{L}]}$
$e_\theta \in$
$R^{V \times D}$
V
D
$T_\theta :$
$R^{N \times L \times D} \rightarrow$
$R^{N \times L \times D}$
$\theta \in$
R^P
$X_{\mathcal{S}} =$
$(x_{i,j})_{i,j \in \mathcal{S}}$
$(\tilde{x}_{i,j})$
e_θ
T_θ
$X_{\mathcal{S}}$
$(\tilde{x}_{i,j})$
T_θ
e_θ
T_θ
$X_{\mathcal{S}}$
e_θ
e_θ^T
T_θ
R^V
$i \in$
$[1, N]$

$$\hat{p}_{i,j} = softmax\left(e_\theta^T \left(T_\theta(e_\theta(\tilde{x}_i))_j\right)\right)$$

$\hat{p}_{i,j}$
j

$$\mathcal{L}(\theta, X, \tilde{X}) = -\frac{1}{|\mathcal{S}|} \sum_{i,j \in \mathcal{S}} \mathbf{1}_{x_{i,j}} \cdot \log(\hat{p}_{i,j})$$

e_θ^T
$e_\theta(x_{i,j})$
$o_{i,j}^\theta =$
$T_\theta(e_\theta(\tilde{x}_i))_j$
e_θ^T
$\mathcal{S}$

$$\mathcal{L}_c(\theta, X, \tilde{X}) = -\frac{1}{|\mathcal{S}|} \sum_{i,j \in \mathcal{S}} \frac{e^{o_{i,j}^\theta \cdot e_\theta(x_{i,j})}}{\sum_{k,l \in \mathcal{S}} e^{o_{i,j}^\theta \cdot e_\theta(x_{k,l})}}$$

Con-
trastive
Weight
Ty-
ing
per
se
do
not
com-
bine
R^V

$|\mathcal{S}| =$
K
$O(KDV)$
$O(K^2D)$
$_2$
$K <$
V
e_θ^T
$K \times$
V
$K \times$
N
$K <$
V

$_2/headless/imgs/bert_train_speed_p13.pngTraininglatency$
$_2/headless/imgs/bert_memory_use_p13.pngMemoryuse$

3
$_{e}ncoder, wetrainavanillaMLMandaheadlesscounterpart.However, wesharetraininghyperparameterssuchasbatchsizea$
$_{m}ulti_{e}nc.$
Fine-tuned on English only
$\pm$
48.0657.3274.0362.72 62 45.2552.1557.36$\pm$
Fine-tuned on target language
$\pm$
54.2566.9573.9669.1467.2260.0467.2265.54$\pm$
$_{c}urve_{m}ultilm.Wefindthattheheadlessapproachleadstosignificantlybetterperformanceforeverylanguageinbothcross-$
$lingualtransferandlanguage-$
$specificfine-$
$tuning.Inaverage, theheadlessMLMoutperformsitsvanillacounterpartby2accuracypointsinthecross-$
$lingualscenario, and by2.7pointsinthelanguage-$
$specificfine-$
$tuningexperiments.$
$_{2}/headless/imgs/xnli_{t}ranslate_{t}rain_{p}13.pngTranslate-Train: targetlanguagefine-tuning$
$_{2}/headless/imgs/xnli_{c}ross_{p}13.pngTranslate-Test: Englishfine-tuning$
$_{c}urve_{m}ultilm, weevaluatethemodelsatintermediatepretrainingcheckpointsandplottheXNLIaveragescoreasafunctiono$
$_{e}\theta$
$_{t}rain_{m}ono_{e}nc.Foreachvocabularysize, wetrainaBPEtokenizersimilartotheBERTtokenizer, andpretrainavanillaMLM$
$_{2}/headless/imgs/bert_{v}ocabsize_{p}13.pngGLUEaveragescore$
$_{2}/headless/imgs/speed_{v}ocabsize_{p}13.pngTrainingspeed$
$_{t}okencountdemonstratesthatincreasingthevocabularysizecomesatalmostnodecreaseintrainingspeedfortheHLMs, cont$
$_{t}rain_{m}ono_{d}ec.HLMsarefine-$
$tunedsimilarlytosec: mono_{d}ecoder.$
$_{2}/headless/imgs/batch_{s}ize_{h}ours_{p}13.pngLAMBADAaccuracyalongpretrainingfordifferentbatchsizes.$
$_{s}ize, weobservethatincreasingbatchsizeleadstobetterperformanceforourHLMs.Whilesmallerbatchsizestraineven fast$
$_{t}rain_{m}ono_{e}ncfordifferentobjectives.Weobservethatadding Cross-$
$EntropytoCWTleadstoslightlyworseperformance, atthecostofreducedthroughput.Wealsonoticethatusingacontrastive$
$\uparrow$
x2.47
x2.1383.37
e
”But what is the difference with softmax?” headless modeling mostly pushes for discrimination between co-occurring tokens
$_{i}nput_{e}mbs.$
$_{2}/headless/imgs/input_{e}mbs_{c}osines_{b}ert.pngMonolingualencoders$
$_{2}/headless/imgs/input_{e}mbs_{c}osines_{p}ythia.pngMonolingualdecoders$
$_{i}nput_{e}mbs, weobservethatHLMstendtogenerallyrepresentsynonymsinamoresimilarwaythanvanillaLMs, ascosine-$
$similaritydistributionsslightlydrifttowardshighervalues.Inaverage, cosine-$
$similaritybetweensynonymsis1.4pointshigherfortheencoderandroughly7pointshigherforboththeoriginalHLMdecoder$
$tunedversion.$
`bert-base-uncased`
`pythia-70m-deduped`
`model-name`
`pythia-70m-deduped`
`pythia-70m-deduped`
ϵ
(β_1, β_2)
`distilbert-base-multilingual-cased`
`distilbert-base-multilingual-cased`
`google/bert_uncased_L-4_H-512_A-8`
`google/bert_uncased_L-4_H-512_A-8`
`gpt2`
`gpt2`
$_{s}ize.Thesemodelsrelyonthe GPT-$
$2architecturewithafewchanges.Thesechangesscaledownthemodelsizeto11Mparameters.$
$(w_c)_{c\in[1,C]}$
$\mathcal{L}_{ce}$

$$\mathcal{L}_{ce}(X,Y) = \sum_{c=1}^{C} \mathcal{L}_c(X,Y)$$

$$\mathcal{L}_c(X,Y) = \sum_{i=1}^{N} \mathbf{1}_{y_i=c} \cdot \mathcal{L}_{ce}(x_i, y_i)$$