Beyond Canonical Fine-tuning: Leveraging Hybrid Multi-Layer Pooled Representations of BERT for Automated Essay Scoring

LREC-COLING 2024

Eujene Nikka Boquio, Prospero Naval, PhD

evboquio@up.edu.ph, pcnaval@dcs.up.edu.ph

University of the Philippines Diliman

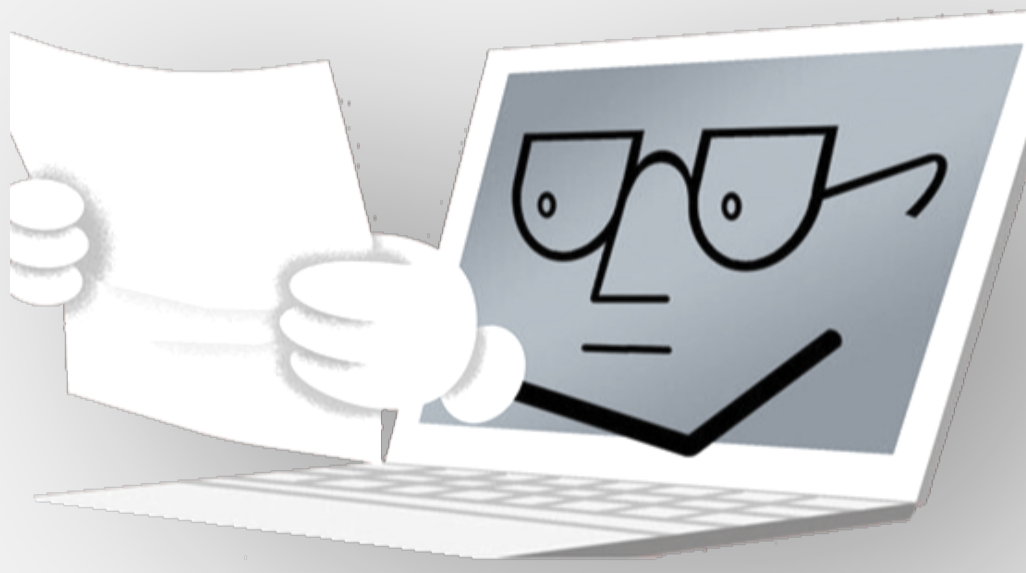
Introduction

Automated Essay Scoring

aims to assign a **numeric score** to an essay written on a certain **topic** or **prompt** based on its **overall quality** or different writing criteria

Challenges

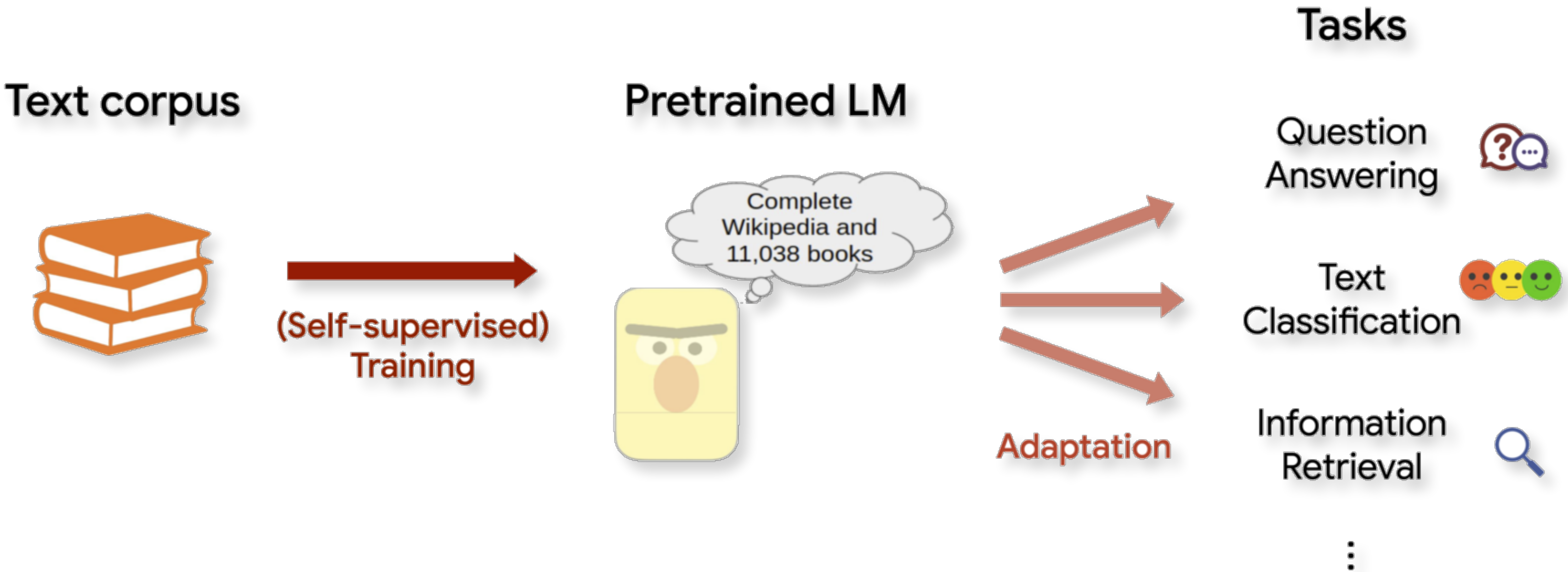
- **lack** of a set of **well-defined standards** or rules for evaluating essays
- essay type and prompt, scoring scale, rubrics, and grade-level of students may **vary**
- essays consist of **long sequences** of words and sentences
- need to consider **higher level features**: semantics, discourse, pragmatics and coherence or adherence to prompt, etc



Introduction

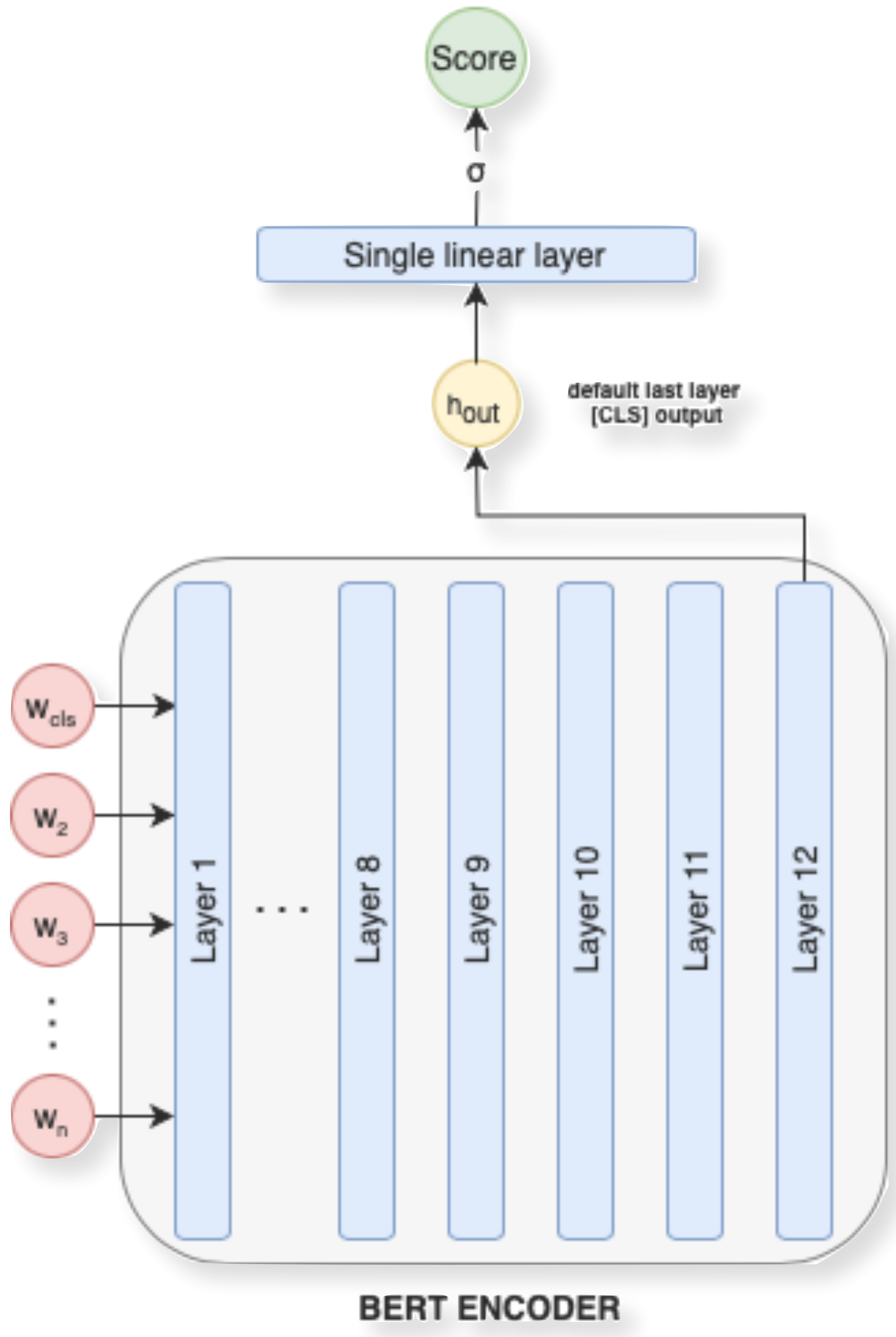
Pre-trained Language Models

- **Transfer learning** allows a **pre-trained** model to be **adapted**
  -> eliminates need to build and train new models from scratch
- PLMs **pre-train** on a large corpus of data and use **transfer learning** for downstream tasks

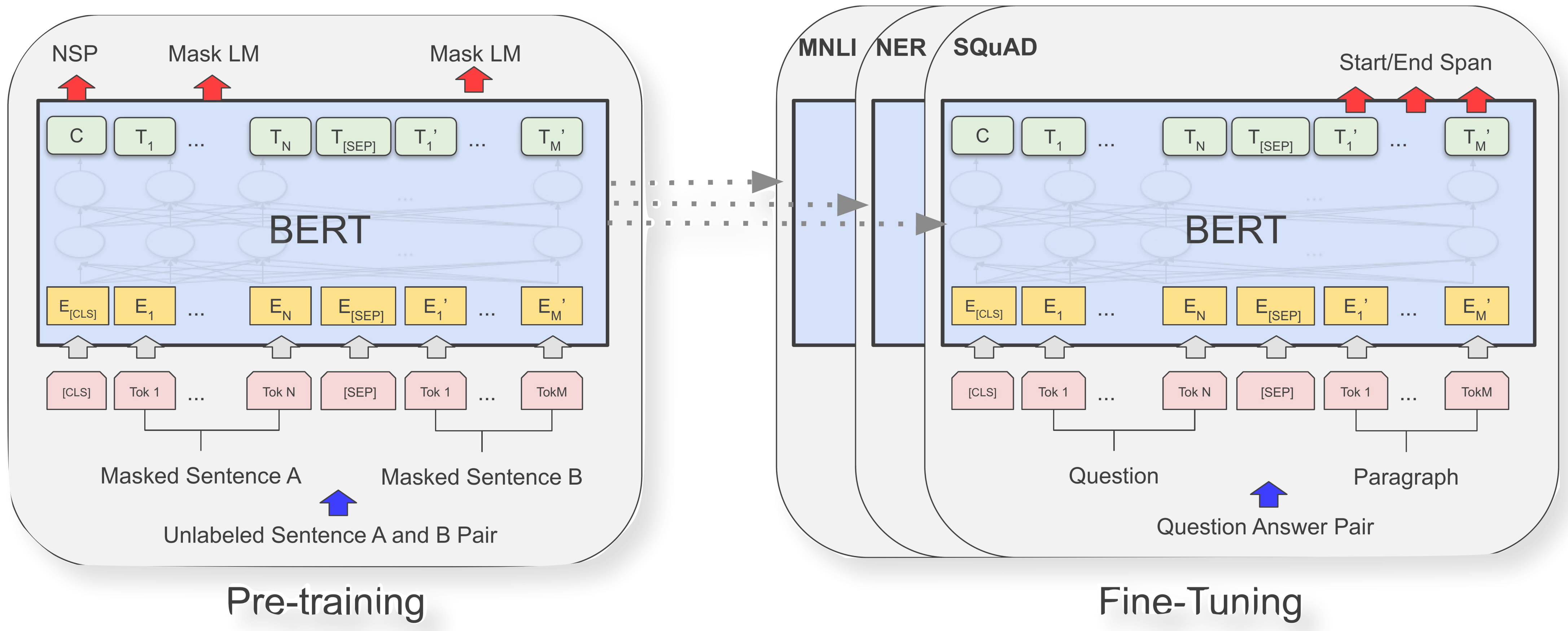


Introduction

Fine-tuning PLMs

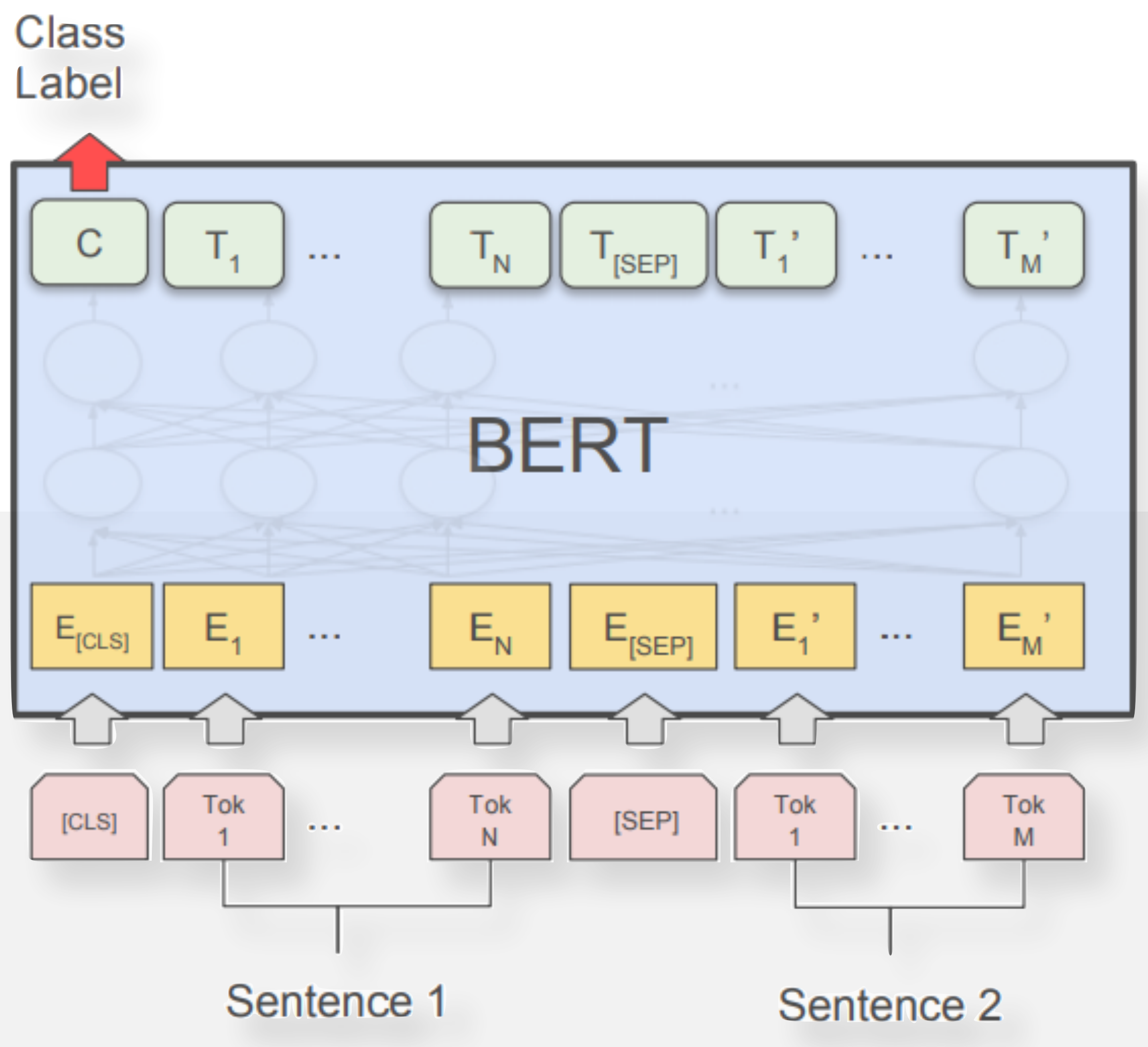


- traditional way of fine-tuning: replace the output layer of the model with a task-specific layer
- pre-training-then-fine-tuning paradigm has since become the common practice in the field of NLP



Devlin et al., 2018

Introduction



BERT

Bidirectional Encoder Representations from Transformers

Masked Language Modeling

- use of a novel **language modeling** approach and a multi-layer **bidirectional Transformer**
- **self-attention mechanism** allows it to capture longer time dependencies and **context**

Next Sentence Prediction

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding

Introduction

Related Studies

- Current AES methods that fine-tune BERT use the output of the final classification ([CLS]) token as essay representation (Rodriguez et al., 2019; Yang et al., 2020; Sun et al., 2022)
- In BERT, the information captured specializes for the language modeling tasks as we approach its last layers (Hao et al., 2020; Peters et al., 2018; Liu et al., 2019).

Devlin et al., 2018

used various combinations of features from different layers for named entity recognition task

Song et al., 2020

used multi-layer representations of the [CLS] token integrated with LSTM and attention pooling for sentiment analysis and NLI tasks

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding

Youwei Song, Jiahai Wang, Zhiwei Liang, Zhiyue Liu, and Tao Jiang. 2020. Utilizing bert intermediate layers for aspect based sentiment analysis and natural language inference.

Ganesh Jawahar, Benoˆıt Sagot, and Djame´ Seddah. 2019. What does bert learn about the structure of language? In ACL 2019-57th Annual Meeting of the Association for Computational Linguistics.

Introduction

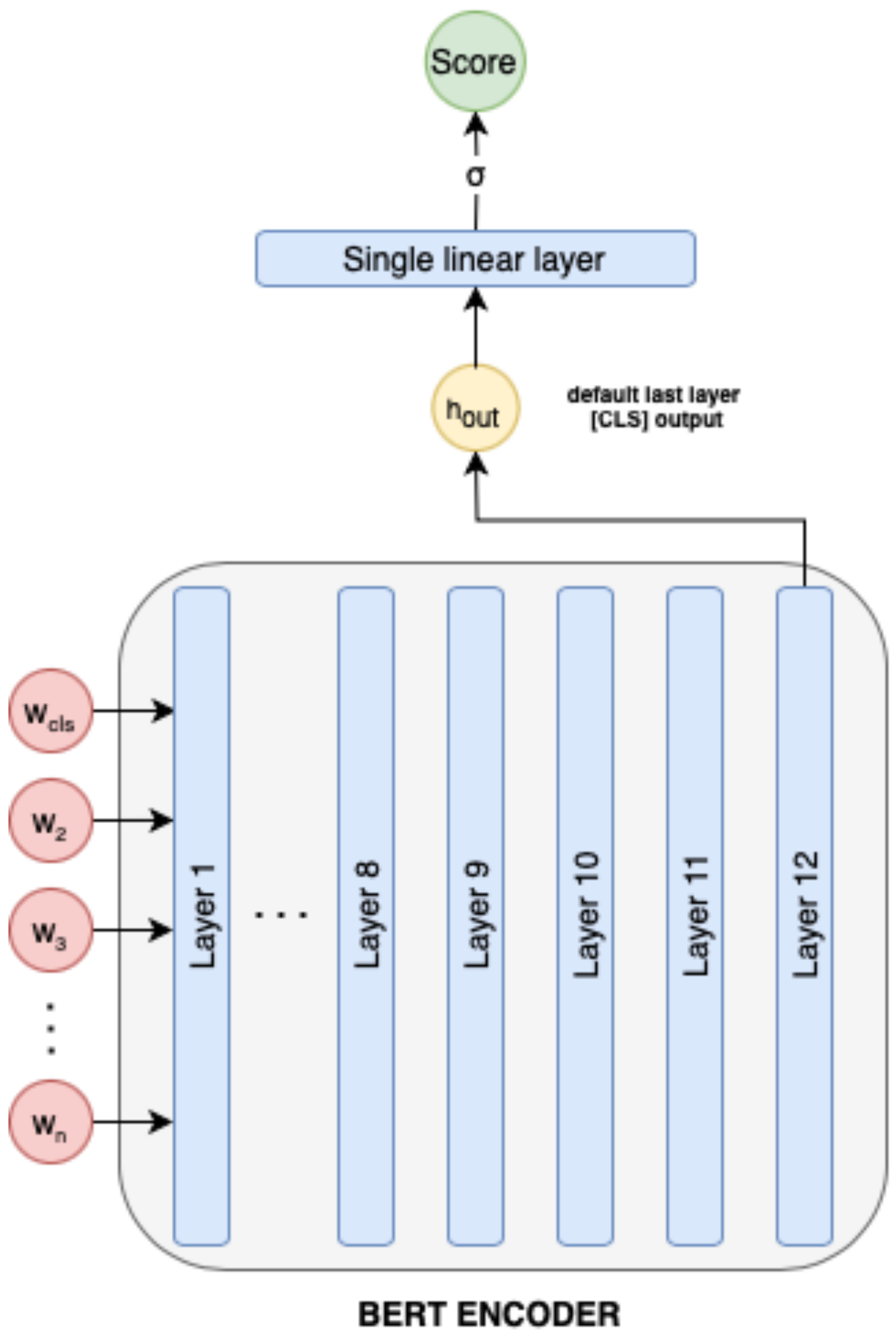
Contributions

first study to go beyond the traditional way of fine- tuning for the **AES task**

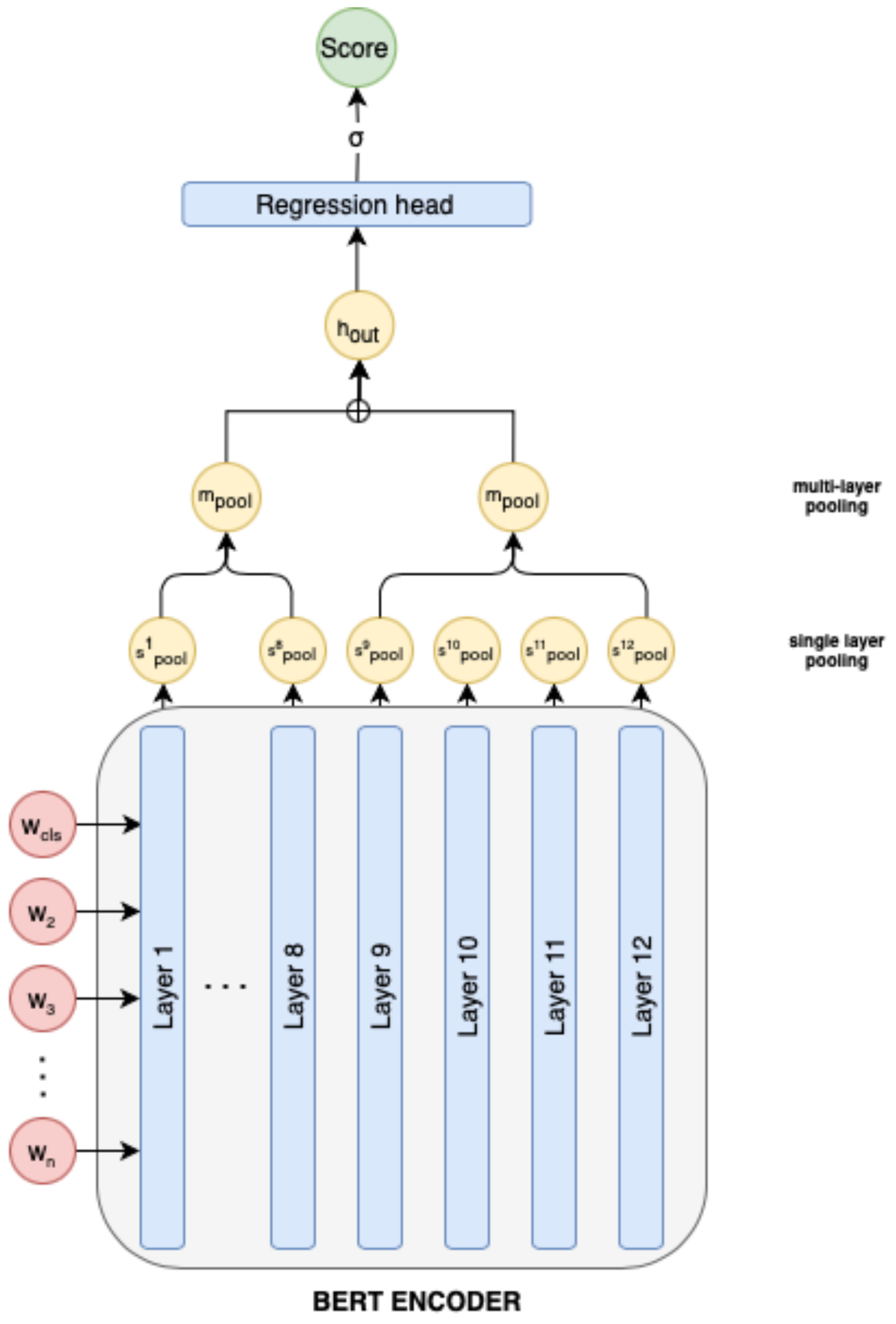
examine the potential of utilizing **BERT inter- mediate layers** and **pooling strategies**

improved results using **pooled** information from **all** BERT layers & **simple** architecture modification

Methodology



(a) Traditional BERT finetuning paradigm



(b) Our proposed model methodology

Methodology

Dataset

Automated Student Assessment Prize (ASAP) dataset

- official dataset used in the ASAP competition in 2012
- 13,000 English essays
- written by students in Grades 7 to 10
- across 8 different prompts, 3 essay types, and different score ranges

Set	Essay Type	# Essay	Ave Len	Score Range	Token Len
1	Argumentative	1785	350	2 - 12	649
2	Argumentative	1800	350	1 - 6	704
3	Source-dependent	1726	150	0 - 3	219
4	Source-dependent	1772	150	0 - 3	203
5	Source-dependent	1805	150	0 - 4	258
6	Source-dependent	1800	150	0 - 4	289
7	Narrative	1569	300	0 - 30	371
8	Narrative	723	650	0 - 60	1077

Methodology

Preprocessing
- convert all characters to **lowercase**
- remove special **characters**
- perform **tokenization** using WordPiece tokenizer
- **truncate** essays longer than 510 tokens
- **pad** shorter essays

AES task
- treat as a **regression** problem
- use **sigmoid activation** function
- use **Mean Square Error** as loss function
- **normalize** reference scores and **scale back** predicted scores to original range

EXPERIMENTAL SETUP

Model Implementation
- **bert-base-uncased** model (**12** layers: hidden size of 768 and 12 attention heads)
- implementation of **Huggingface** transformers library

Training and Evaluation
- **train/validation/test split** of 60/20/20
- **quadratic weighted kappa** (QWK) as evaluation metric
- train models **100 epochs**
- choose model w/ **best validation QWK**
- implemented using **PyTorch**

Methodology

MAIN EXPERIMENTS

INITIAL EXPERIMENTS

Using default lhs-cls outputs & single output layer

Hyperparameters	**Values tested**
Optimizer	**SGD**, Adam, **AdamW**
Scheduler	None, **Linear decay**, Cosine, Polynomial decay
Learning rate	SGD: 0.001, **0.01**, **0.03**, 0.05, 0.08, 0.1, Adam/AdamW: $3e^-6$, e^-5, $5e^-5$, $1e^-4$
Dropout rate	0, **0.1**, **0.3**, 0.5

REPRESENTATION LEARNING

MODEL OUTPUTS

explore two main outputs of BERT

POOLING STRATEGIES

single- and multi-layer pooling strategies

MODEL ARCHITECTURES

MODIFICATIONS

3 model architectures with different task-specific component

Methodology

Representation Learning

BERT's default output representation:

Given a sequence of n tokens $\{w_1, \ldots, w_n\}$, which include special tokens and the words in an input essay, BERT encodes the sequence into the contextualized representation $R \in R^{n \times d}$ given by:

$$R = BERT(\{w_1, \ldots, w_n\})$$

where R is the output of the last layer of the BERT encoder and d is the hidden size. R corresponds to the first token ([CLS]) of the last hidden state.

2 MAIN BERT OUTPUTS

last hidden state (lhs)

sequence of hidden states at the last layer of the model

Notations:

- lhs - last layer
- 2lhs - 2nd to the last layer
- 3lhs - 3rd to the last layer
- 4lhs - 4th to the last layer

hidden state (hs)

aggregation of hidden states of multiple layers and sequences

Notations:

- gl4 - last 4 layers (get last 4)
- gf8 - first 8 layers (get first 8)
- ahs - all 12 layers (all hidden states)

Methodology

Single Layer Pooling Strategies

CLS embedding (cls)

obtained by taking h^K_{cls} of a layer K. e.g. the CLS embedding of the third-to-the last layer (3lhs-cls) is h^{10}_{cls}

Mean pooling (mean)

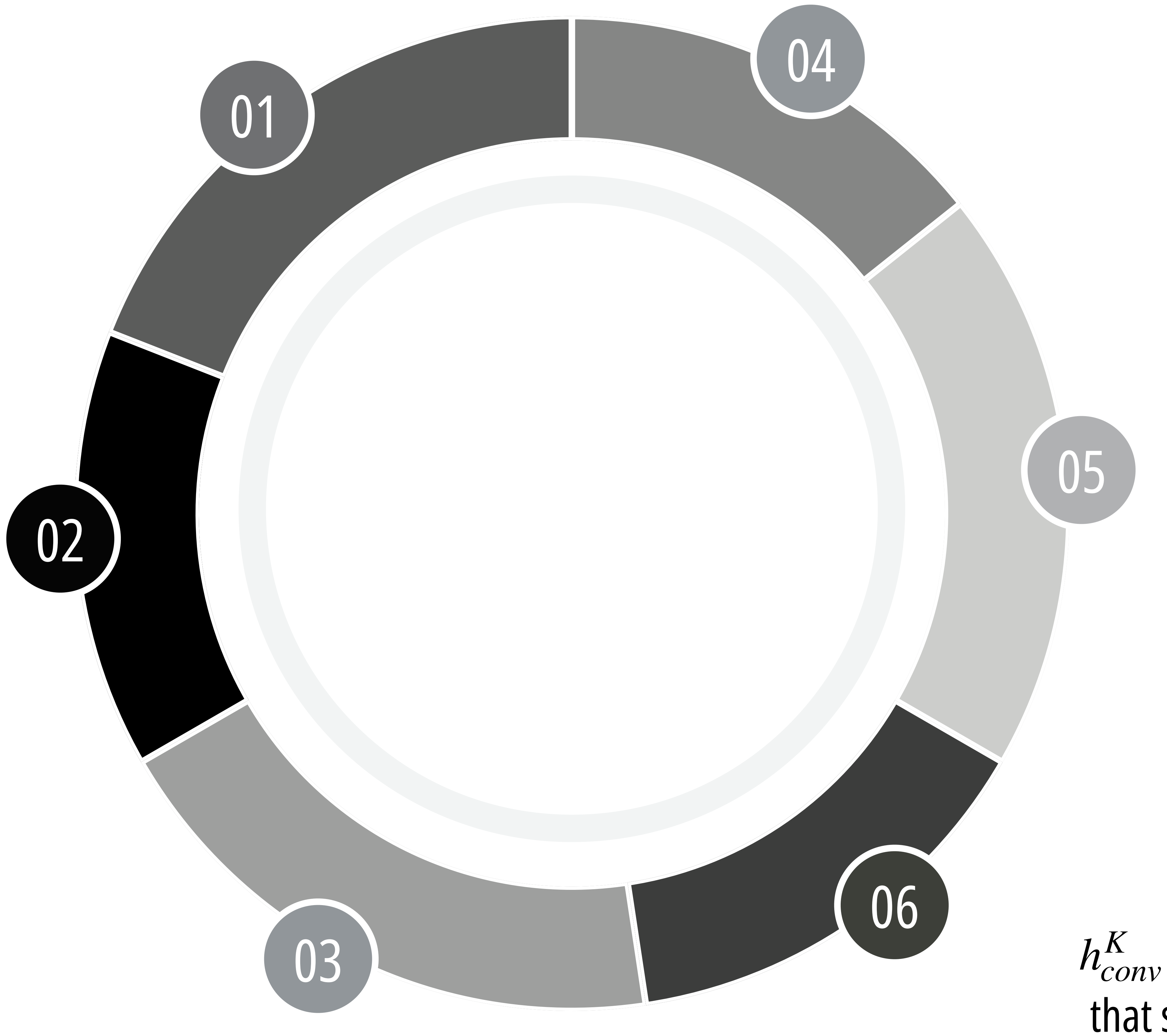
averages the hidden states of all n token embeddings in a layer. We ignore the [PAD] token by utilizing the attention masks.

$$\mathrm{h}^K_{mean} = 1n \sum_{i=1}^{n} H^K_i$$

Max pooling (max)

takes the maximum across n token embeddings. Attention masks are also used.

$$\mathrm{h}^K_{max} = \max_{i=1,\ldots,n} H^K_i$$



Mean-max pooling (mm)

finds both mean and max pooling embeddings and concatenates them

$$\mathrm{h}^K_{mm} = h^K_{mean} \parallel h^K_{max}$$

Attention pooling (att)

uses dot-product attention operation on all token embeddings for a layer K

$$\mathrm{a}_i = \tanh\left(W^K_a \cdot h^K_i + b_a\right)$$
$$\alpha_i = e^{w_\alpha \cdot a_j} \sum e^{w_\alpha \cdot a_j}$$
$$\mathrm{h}^K_{att} = \sum \alpha_{i_i}^{K}$$

where W_a is the weight matrix, w_α is the weight vector, b_a is the bias vector, and a_i and α_i are the attention vector and attention weight for the i^{th} token respectively.

Conv1d pooling (conv)

h^K_{conv} uses 1D convolution layers (Kiranyaz et al., 2021) that slide across all n tokens. We use a kernel size of 2 tokens and a padding size of 1.

Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 2021. 1d convolutional neural networks and appli- cations: A survey. Mechanical systems and signal processing, 151:107398.

Methodology

Multi-layer Pooling Strategies

We denote the hidden states of the CLS token of BERT with L layers as $h_{CLS} = h^1_{CLS}, h^2_{CLS}, \ldots, h^L_{CLS}$.

Mean pooling (mean-hs)

takes the mean pooling of the outputs for each layer in a combination of layers and stacking them together

Concatenate pooling (concat)

concatenates outputs from multiple layers into one. e.g. gl4-concat concatenates the outputs from the last 4 layers.

Weighted layer pooling (wl)

takes the weighted mean of the token embeddings of layers in a set of layers

By default, pooling strategies for hs outputs are applied on the CLS embeddings from a set of layers **S**

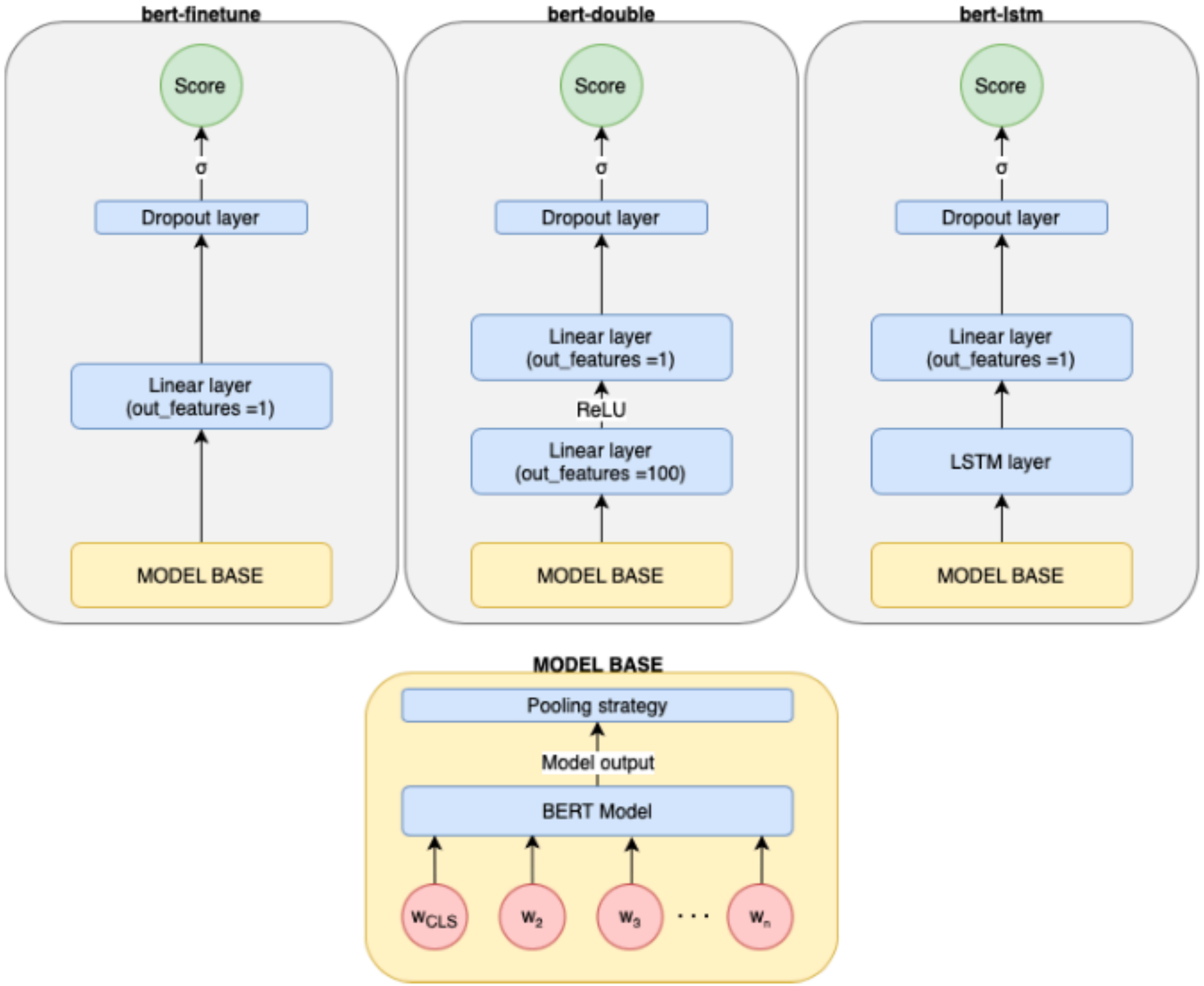
(e.g. gl4-concat uses concatenates the CLS embeddings of the last 4 layers, or S = {9, 10, 11, 12}).

When using single-layer pooling other than CLS embedding, we add the pooling method after the notation

(e.g. gl4-att-concat concatenates output obtained from attention pooling from each of the last 4 layers)

Methodology

Model Architectures



Default Configuration

- batch size of 32
- dropout rates of 0, 0.1, and 0.3
- linear scheduler w/ 0 warmup steps
- gradient clipping w/ max norm of 1.0
- initialize model parameters to the pre-trained values

Results and Discussion

Initial Experiments

MODEL	CONFIG	SET 1	SET 2	SET 3	SET 4	SET 5	SET 6	SET 7	SET 8	AVE
bert-finetune	AdamW lr=3e^{-6}	**0.8480**	0.6361	0.6774	0.8085	0.8024	0.8179	0.8022	0.7678	0.7700
	AdamW lr=5e^{-5}	0.8216	**0.6579**	0.6888	0.8135	0.7957	0.8228	0.8114	0.7007	0.7641
	AdamW lr=1e^{-4}	0.8279	0.6568	0.6607	0.8232	**0.8236**	0.8191	**0.8141**	0.6793	0.7631
	SGD lr=0.01	0.8351	0.6454	**0.6977**	**0.8298**	0.7999	0.8235	0.7991	**0.7754**	**0.7757**
	SGD lr=0.03	0.8152	0.6000	0.6913	**0.8298**	0.8193	**0.8309**	0.8068	0.7386	0.7665
bert-double	AdamW lr=3e^{-6}	0.8186	0.6262	0.6844	0.7925	0.7680	0.8054	0.7989	0.7501	0.7555
	AdamW lr=5e^{-5}	0.8106	**0.6809**	0.6744	0.8134	**0.8173**	0.8208	0.8131	0.7097	0.7675
	AdamW lr=1e^{-4}	0.8130	0.6583	0.7075	0.8127	0.8155	0.8233	0.7915	0.6887	0.7638
	SGD lr=0.01	**0.8277**	0.6400	**0.7096**	0.7975	0.7924	0.8134	0.8107	0.7591	0.7688
	SGD lr=0.03	0.8166	0.6608	0.7039	**0.8168**	0.8095	**0.8353**	**0.8193**	**0.7689**	**0.7789**
bert-lstm	AdamW lr=3e^{-6}	0.8207	0.6051	0.7031	0.7809	0.7824	0.8227	0.7985	0.7343	0.7560
	AdamW lr=5e^{-5}	0.8022	0.6681	0.6811	0.8239	0.7949	**0.8436**	0.8044	0.7284	0.7683
	AdamW lr=1e^{-4}	0.8233	0.6303	**0.7105**	0.8190	**0.8129**	0.8268	0.7950	0.7429	0.7701
	SGD lr=0.01	**0.8259**	0.6524	0.6947	0.8133	0.8078	0.8108	0.8066	**0.7505**	0.7702
	SGD lr=0.03	0.8202	**0.6803**	0.7070	**0.8271**	0.8029	0.8269	**0.8124**	0.7468	**0.7780**

QWK scores for different model architectures and hyperparameter configuration on ASAP dataset. The default lhs-cls output is used for all models.

SGD optimizer has better scoring performance on all essay prompts using all 3 model architectures

Results and Discussion

Main Results

MODEL	POOL	lhs	2lhs	3lhs	4lhs	AVE
bert-finetune	**cls**	0.8152	0.8084	0.8243	**0.8234**	0.8178
	att	**0.8398**	0.8155	**0.8377**	0.8071	**0.8250**
	mean	0.8205	**0.8358**	0.8261	0.8154	0.8245
	max	0.8333	0.8111	0.7673	0.8163	0.8070
	mm	0.8240	0.8208	0.8177	0.7490	0.8029
	conv	0.8015	0.7856	0.7825	0.7906	0.7901
bert-double	**cls**	0.8202	0.8154	0.8027	0.8259	0.8160
	att	0.8231	0.8176	0.8123	0.8272	0.8201
	mean	0.8238	0.8157	0.8182	0.8239	0.8204
	max	0.8351	0.8358	0.8215	**0.8274**	0.8299
	mm	0.8206	**0.8419**	**0.8376**	0.8169	0.8293
	conv	**0.8381**	0.8380	0.8239	0.8263	**0.8316**
bert-lstm	**cls**	0.8202	0.7983	0.8186	0.8273	0.8161
	att	0.8183	0.8137	**0.8302**	0.8083	0.8176
	mean	0.8349	0.8263	0.8089	0.8280	0.8245
	max	0.8349	**0.8444**	**0.8301**	0.8169	0.8316
	mm	**0.8446**	0.8309	**0.8302**	**0.8361**	**0.8354**
	conv	0.8321	0.8103	0.8084	0.8299	0.8202

QWK scores using different pooling strategies on each of the last 4 layers. Best values for each model and layer are in bold.

- for **bert-finetune**, the CLS output from only the last layer may not be the best essay representation
- for **bert-double**, **all** other pooling methods **improved** over the cls embedding
- for **bert-lstm**, **all** other pooling methods **improved** over the cls embedding
- modified model architectures and other pooling methods **improved** over the **default** BERT fine-tuning

Results and Discussion

Main Results

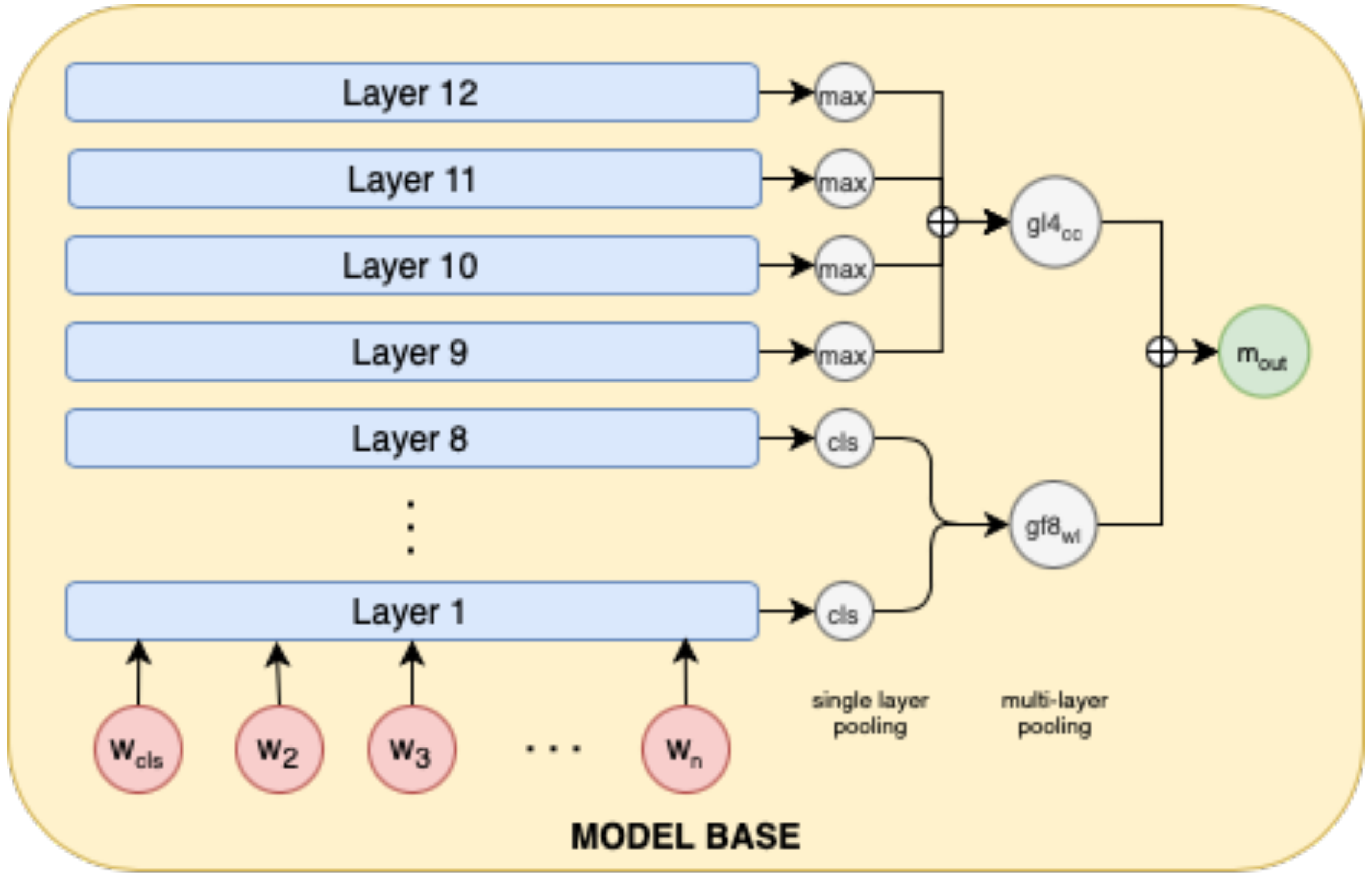
- **all layers** contain important essay information that can still help with the scoring performance
- **first layers** can still contribute to obtaining relevant essay representations
- observe **bert-double** to be the best at capturing relevant information from combinations of layers
- observe the best QWK results using **gl4-cc** and **gf8-wl**

MODEL	POOL	gl4	gf8	ahs
bert-finetune	mean-hs	0.8118	0.8174	0.8127
	cc	0.8127	**0.8323**	0.8164
	wl	**0.8207**	0.8270	**0.8361**
bert-double	mean-hs	0.8288	0.8163	0.8311
	cc	**0.8445**	0.8281	**0.8388**
	wl	0.8297	**0.8385**	0.8253
bert-lstm	mean-hs	**0.8265**	0.8035	0.8063
	cc	0.8158	-	-
	wl	0.8190	**0.8340**	**0.8257**

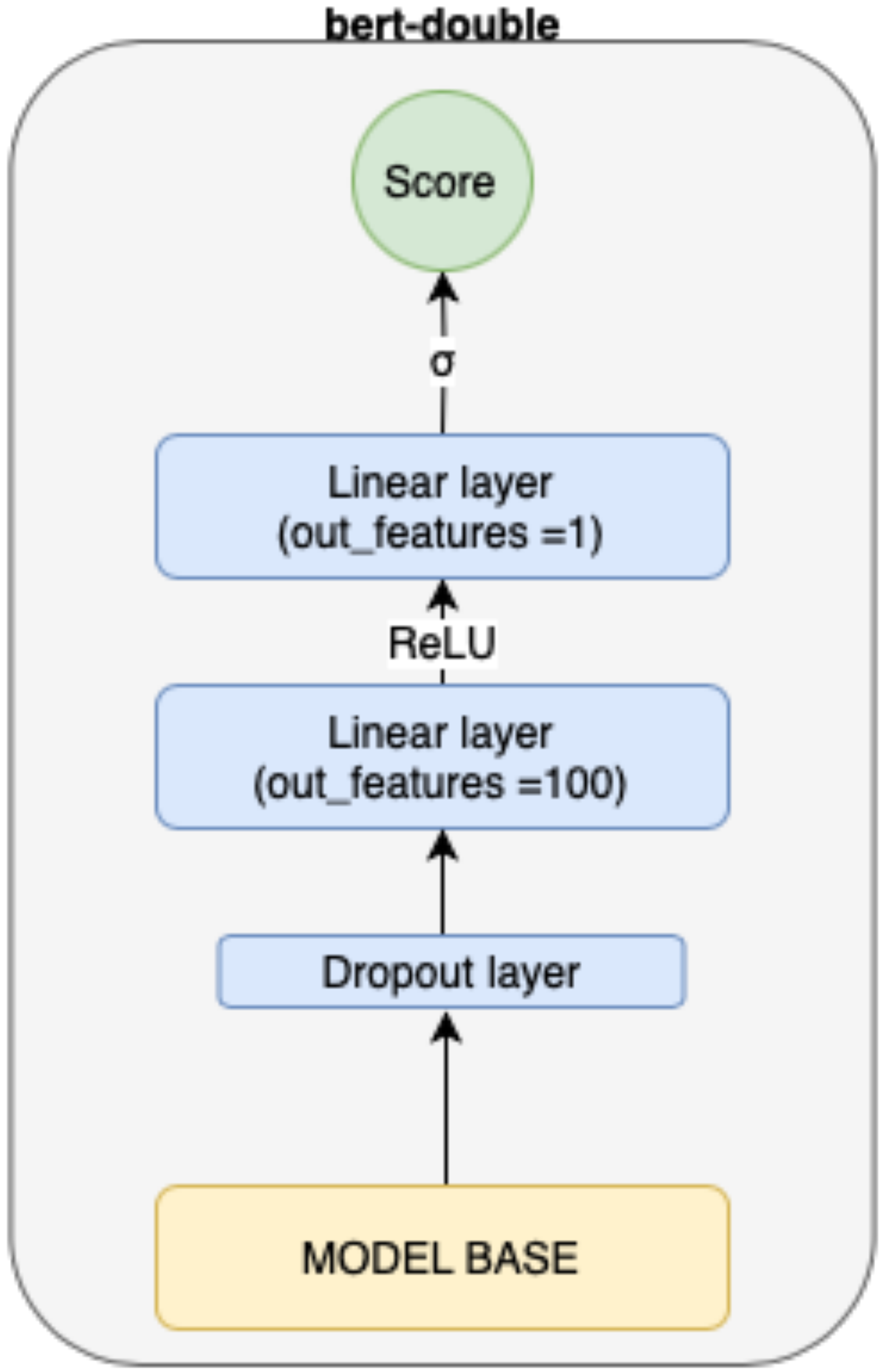
QWK scores for different pooling strategies on different layer combinations. Best values for each model and layer combination are in bold.

Results and Discussion

Main Results



ahs-cls-wlcc
ahs-max-wlcc



Results and Discussion

Main Results

OUTPUT	SET 1	SET 2	SET 3	SET 4	SET 5	SET 6	SET 7	SET 8	AVE
lhs-cls	0.8277	0.6400	0.7096	0.7975	0.7924	0.8134	0.8107	0.7591	0.7688
gl4-cc	**0.8445**	0.6488	0.7054	0.8099	0.8053	0.8096	0.8088	0.7567	0.7736
gf8-wl	0.8385	0.6779	0.7040	0.8030	0.8009	0.8143	**0.8194**	0.7069	0.7706
ahs-cls-wlcc	0.8293	0.6737	0.7036	**0.8316**	0.7936	0.8309	0.8040	0.7638	0.7822
ahs-att-wlcc	0.8351	0.6746	0.7168	0.8131	0.8092	0.8059	0.8089	**0.7865**	0.7813
ahs-mean-wlcc	0.8392	0.6783	0.6958	0.8186	0.8041	**0.8380**	0.8081	0.7549	0.7796
ahs-max-wlcc	0.8338	0.6884	0.7080	0.8214	**0.8320**	0.8277	0.8172	0.7714	**0.7875**
ahs-mm-wlcc	0.8406	0.6618	**0.7318**	0.8106	0.8098	0.8240	0.8120	0.7658	0.7820
ahs-conv-wlcc	0.8168	**0.7288**	0.6945	0.8097	0.8166	0.8024	0.8100	0.7545	0.7791

QWK scores of our model using different outputs on ASAP dataset. The best values for each essay set are shown in bold.

- **all other model outputs** improved over default lhs-cls output
- using **all 12 layers** performs **better** than only using a subset of layers
- **all** the models that use **hybrid pooling** have an improvement in average QWK scores
- best average QWK score is **ahs-max-wlcc**, followed by **ahs-cls-wlcc**

Results and Discussion

Set	Essay Type	# Essay	Ave Len	Score Range	Token Len
1	Argumentative	1785	350	2 - 12	649
2	Argumentative	1800	350	1 - 6	704
3	Source-dependent	1726	150	0 - 3	219
4	Source-dependent	1772	150	0 - 3	203
5	Source-dependent	1805	150	0 - 4	258
6	Source-dependent	1800	150	0 - 4	289
7	Narrative	1569	300	0 - 30	371
8	Narrative	723	650	0 - 60	1077

Comparisons with Baselines

MODEL	SET 1	SET 2	SET 3	SET 4	SET 5	SET 6	SET 7	SET 8	AVE
EASE (SVR)	0.781	0.621	0.63	0.749	0.782	0.771	0.727	0.534	0.699
EASE (BLRR)	0.761	0.606	0.621	0.742	0.784	0.775	0.73	0.617	0.705
LSTM	0.775	0.687	0.683	0.795	0.818	0.813	0.805	0.594	0.746
LSTM + CNN	0.821	0.688	0.694	0.805	0.807	0.819	0.808	0.644	0.76
BERT	0.792	0.679	0.715	0.8	0.805	0.805	0.785	0.595	0.748
XLNet	0.776	0.68	0.692	0.806	0.783	0.793	0.786	0.628	0.743
BERT Ensemble	0.802	0.672	0.708	0.815	0.806	0.814	0.804	0.597	0.752
XLNet Ensemble	0.804	0.685	0.7009	0.795	0.799	0.805	0.8	0.597	0.748
BERT + XLNet Ensemble	0.807	0.696	0.703	0.819	0.808	0.815	0.806	0.604	0.757
HISK and -SVR	0.836	0.724	0.677	0.821	0.83	0.828	0.801	0.726	0.78
BOSWE and -SVR	0.788	0.689	0.667	0.809	0.824	0.824	0.766	0.679	0.756
HISK+BOSWE and -SVR	**0.845**	0.729	0.684	0.829	0.833	0.83	0.804	0.729	0.784
Parameter-Efficient Transformer	0.743	0.674	0.718	**0.884**	0.834	0.842	0.819	0.744	0.785
HA-LSTM+SST+DAT	0.836	**0.73**	**0.732**	0.822	0.835	0.832	0.821	0.718	0.79
BERT+SST+DAT	0.824	0.699	0.726	0.859	0.822	0.828	**0.84**	0.726	0.791
$R^2 BERT$	0.817	0.719	0.698	0.845	**0.841**	**0.847**	0.839	0.744	**0.794**
BERT-ahs-cls-wlcc (ours)	0.8293	0.6737	0.7036	0.8316	0.7936	0.8309	0.8040	0.7638	0.7822
BERT-ahs-wm-wlcc (ours)	0.8338	0.6884	0.7080	0.8214	0.8320	0.8277	0.8172	**0.7714**	**0.7875***

QWK scores of our chosen models and other baseline models on the ASAP dataset.
Best values for each essay set are in bold. Models that outperform ours use self-supervised tasks, multi-loss learning functions, or more intricate architectures.

Summary

Our model

- simple model modification using a hybrid pooled multi-layer BERT representation

utilize **intermediate layers** & different ways of pooling each single layer & multiple layers for the **fine-tuning** of **BERT** for AES

found that we can **improve** essay scoring performance by using **all** or even **several BERT layers**

improved results with **simple** architectural modification of the task-specific layer with **ReLU**

performed **hyperparameter tuning** and found that **SGD** optimizer generalizes **better** than Adam for AES using BERT

Future Work

OTHER LONG-TEXT TASKS

evaluate across various datasets and other long-text tasks

ERROR ANALYSIS

to detect future improvements

DIFFERENT AES SETTINGS

evaluate on trait-specific scoring and cross- prompt settings

MORE COMPLICATED TECHNIQUES

more complicated architectures, multitask learning with other loss functions or NLP tasks, and various optimization strategies

Thank you!

Contact

Nikka Boquio

evboquio@up.edu.ph

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding

Ganesh Jawahar, Benoˆıt Sagot, and Djame´ Seddah. 2019. What does bert learn about the structure of language? In ACL 2019-57th Annual Meeting of the Association for Computational Linguistics.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2020. In- vestigating learning dynamics of bert fine-tuning. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Lin- guistics and the 10th International Joint Conference on Natural Language Processing, pages 87–92.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019. Lin- guistic knowledge and transferability of contextual representations. arXiv preprint arXiv:1903.08855.

Matthew E Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. arXiv preprint arXiv:1808.08949.

Pedro Uria Rodriguez, Amir Jafari, and Christopher M Ormerod. 2019. Language models and automated essay scoring. arXiv:1909.09482.

Youwei Song, Jiahai Wang, Zhiwei Liang, Zhiyue Liu, and Tao Jiang. 2020. Utilizing bert intermediate layers for aspect based sentiment analysis and natural language inference.

Jingbo Sun, Tianbao Song, Jihua Song, and Weim- ing Peng. 2022. Improving automated essay scor- ing by prompt prediction and matching. Entropy, 24(9):1206.

Junjie Yang and Hai Zhao. 2019. Deepening hid- den representations from pre-trained language mod- els for natural language understanding. ArXiv abs/ 1911.01940.

Methodology

Baseline Models

EASE (Phandi et al., 2015)

open source system that uses Support Vector Regression (SVR) and Bayesian Linear Ridge Regression (BLRR)

Hybrid models (Cozma et al., 2018)

HISK and ν-SVR, BOSWE and ν-SVR, and HISK+BOSWE and ν-SVR are reported.

LSTM-based deep neural networks (Taghipour and Ng, 2016)

vanilla LSTM network and the LSTM + CNN network that combines CNN and LSTM ensembles over 10 runs

Methodology

Baseline Models

Simpler transformer-based models (Rodriguez et al., 2019)

use BERT and XLNet (Yang et al., 2019) models, as well as ensembles for BERT, XLNet, and their combination

Parameter-Efficient Transformer (Sethi and Singh, 2022)

use transformer-based pre-trained language model with adapter models to reduce number of trainable parameters

Self-supervised methods (Cao et al., 2020)

use two self-supervised tasks and a domain adversarial training technique. use hierarchical LSTM model and BERT as their base encoders, which are HA- LSTM+SST+DAT and BERT+SST+DAT respectively.

R^2BERT (Yang et al., 2020)

employs a multi-loss function that combines regression and ranking to fine-tune BERT model