Does surgical fine-tuning work in NLP and does it reflect the classical NLP pipeline?

Stanford C330 Autumn 2022: Final Project

Pooja Sethi
Department of Computer Science
Stanford University
pjasethi@stanford.edu

Abstract

I investigate whether the new technique of *surgical fine-tuning* (Lee et al., 2022b), which has demonstrated success in adapting pretrained image models to datasets displaying distribution shifts, also provides benefits in standard NLP classification tasks. I evaluate surgical fine-tuning on token-level tasks (POS, NER, and Chunking) from CoNLL-2003 (Tjong Kim Sang & De Meulder, 2003) and sequence-level tasks (Grammaticality) from GLUE (Warstadt et al., 2018; Wang et al., 2018). In my experiments, I test surgical fine-tuning by separately fine-tuning five distinct, but overlapping, blocks of four layers of BERT (Devlin et al., 2019), along with the classification head.

I find that compared to one of the baselines, i.e., widely-used last-layer fine-tuning, surgical fine-tuning provides significant benefits. Although most of the performance benefits appear to be due to fine-tuning an overall larger number of parameters, the location of the parameters does matter as well. Surgical fine-tuning consistently peaked in performance at the center layers of the model (Block 3). Unlike Lee et al. (2022b), however, I found that fine-tuning all model parameters was always best.

The second line of investigation in this work was to understand whether the performance benefits gained by surgical fine-tuning appear in a manner that reflect earlier-discovered phenomena, i.e., that BERT rediscovers a classical NLP pipeline (Tenney et al., 2019) and that it learns a hierarchy of linguistic information, "starting with surface features at the bottom, syntactic features in the middle followed by semantic features at the top" (Jawahar et al., 2019). Lee et al. (2022b) also found that the best choice of block to surgically fine-tune was dependent on the nature of the domain-shift: input-level, feature-level, or output-level.

Surprisingly, I find that surgical fine-tuning did not seem to rediscover the classical NLP pipeline as I had expected, as fine-tuning the middle block consistently led to the best performance across all tasks I tested. Although the reasons for this are mysterious, I have two possible explanations. The first is that I simply did not test surgical fine-tuning on a broad enough spectrum of NLP tasks (particularly surface-level tasks) for the trend to appear. The second, incited by recent work by Niu et al. (2022), is that the inner workings of BERT may be more complex than what can be explained by layer depth alone. Attention-based probing methods may provide greater clarity than performance-based probing methods as to which model parameters are best to surgically fine-tune.

All code is available at:
https://github.com/poojasethi/surgical-ft-nlp.

1 Introduction

A new approach to fine-tuning, called *surgical fine-tuning*, demonstrates that selectively choosing an arbitrary subset of layers to fine-tune can lead to improved performance compared over standard, last layer (or last few layers) fine-tuning of pretrained models (Lee et al., 2022b). The authors empirically demonstrate that this holds true in the context of adapting pretrained image models in benchmarks displaying distribution shifts. Interestingly, Lee et al. (2022b) find that the location of the best block to fine-tune reflects the nature of the domain shift: *input-level*, *feature-level*, or *output-level*. Such domain shifts are illustrated in Figure 1a.

Separately, and in earlier work, Tenney et al. (2019) show that the internal representations learned by BERT (Devlin et al., 2019) follow what they coin the *classical NLP pipeline*. That is, BERT attends more heavily to earlier layers for certain tasks (syntactic information) and more heavily to later layers for others (semantic understanding). This allows us to define a notion of ordering from low-level to high-level NLP tasks. These results are illustrated in Figure 1b.

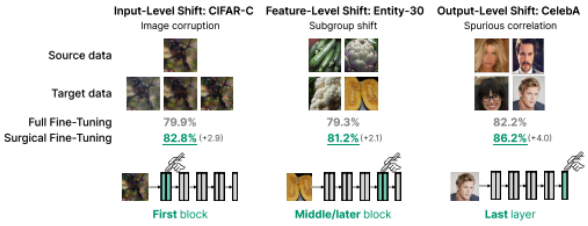


Figure 1: Surgical fine-tuning, where we tune only one block of parameters and freeze the remaining parameters, outperforms full fine-tuning on a range of distribution shifts. Moreover, we find that tuning different blocks performs best for different types of distribution shifts. Fine-tuning the first block works best for input-level shifts such as CIFAR-C (image corruption), later blocks work best for feature-level shifts such as Entity-30 (shift in entity subgroup), and tuning the last layer works best for output-level shifts such as CelebA (spurious correlation between gender and hair color).

(a) An illustration of results from Lee et al. (2022b). Fine-tuning the first block works best for input-level domain shifts such as CIFAR-C (image corruption), while fine-tuning the last block works best for output-level domain shifts such as CelebA (spurious correlation between gender and hair color).

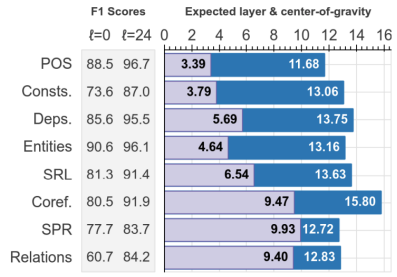


Figure 1: Summary statistics on BERT-large. Columns on left show F1 dev-set scores for the baseline ($P_\tau^{(0)}$) and full-model ($P_\tau^{(L)}$) probes. Dark (blue) are the mixing weight center of gravity (Eq. 2); light (purple) are the expected layer from the cumulative scores (Eq. 4).

(b) An illustration of the classical NLP pipeline from Tenney et al. (2019). We can see that BERT has a relatively low expected layer and center-of-gravity for syntactic tasks like POS-tagging and constituency parsing and a high expected layer for semantic understanding tasks like proto-role labeling and relations. Refer to the original paper for Equations.

Figure 1: A closer look at layer-wise results from prior work.

Connecting these two bodies of work leads to natural questions: does surgical fine-tuning also provide benefits in *NLP tasks*, and if so, does the choice of the best layer to fine-tune reflect the expected order of the task? More concretely, the novel technical contributions of this work are to investigate and begin to answer the following the two questions:

1. Can we empirically show that surgical fine-tuning of a large language model, such as the widely-used BERT model (Devlin et al., 2019), outperforms baselines on standard NLP benchmarks?
2. Do lower-level language tasks (syntactic information) benefit more heavily from surgical fine-tuning of earlier layers, while higher-level language tasks (semantic reasoning) benefit more from fine-tuning of later layers – in other words, do the performance benefits of surgical fine-tuning (if any) appear in a manner that reflect the classical NLP pipeline?

2 Related Work

While Lee et al. (2022b) show that surgical fine-tuning provides benefit when adapting various pretrained vision models to domain shifts, such as ResNet-50 and CLIP ViT-B/16 (Vision Transformer)

(Croce et al., 2020; Radford et al., 2021) , they do not explicitly test the technique on NLP models or tasks. To my knowledge, no other prior work has explicitly demonstrated the utility of surgical fine-tuning in the context of NLP. The work of Tenney et al. (2019) was written prior to the introduction of surgical fine-tuning and thus also does not discuss the technique.

The idea of fine-tuning as a general approach to transfer learning in NLP was introduced by Howard & Ruder (2018) and has been further explored by many. Ruder (2021) categorizes recent advances in NLP fine-tuning into areas such as **adaptive** (Dai & Le, 2015; Howard & Ruder, 2018; Logeswaran et al., 2019; Han & Eisenstein, 2019; Mehri et al., 2019), **behavioral** (Phang et al., 2018; Aghajanyan et al., 2021), **parameter-efficient** (Rebuffi et al., 2017; Houlsby et al., 2019; Stickland & Murray, 2019; Hu et al., 2021), and **text-to-text** fine-tuning (Schick & Schütze, 2020; Brown et al., 2020).

In adaptive and behavioral fine-tuning, an additional intermediate step is performed where the model is fine-tuned on in-distribution data (using the unsupervised pre-training objective) or on related tasks, respectively, before being further fine-tuned in a supervised manner to perform the final target task. An important point to note is that these methods do not necessarily *preclude* surgical fine-tuning, which is an orthogonal technique that can be done independently or in conjunction with adaptive or behavioral fine-tuning. (I study surgical fine-tuning independently.)

In adapters, the most notable method of parameter-efficient fine-tuning, small bottleneck layers are inserted between the layers of a pretrained model (Houlsby et al., 2019). Thus, adapters are similar to surgical fine-tuning in that they allow fine-tuning to occur in arbitrary locations in the pretrained model, but distinct in that they introduce a new set of parameters as opposed to reusing the existing parameters of the pretrained model. Similarly, other parameter-efficient methods, such as Low-rank adaptation of Language Models (LoRA), fine-tune new low-rank residual matrices instead of directly updating the model parameters (Hu et al., 2021).

Finally, in text-to-text fine-tuning, the target task is reframed as the pretraining objective, so no new parameters need to be learned from scratch. This being said, Ruder (2021) still recommends the aforementioned approaches in most practical settings.

More broadly, many approaches to meta-learning for NLP are also discussed by Lee et al. (2022a) in their survey paper. While they do discuss how certain meta-learning techniques, which they categorize as “learning to initialize", can lead to faster fine-tuning convergence, they do not explicitly discuss surgical fine-tuning as a potentially better alternative to standard last-layer fine-tuning.

3 Approach

Surgical fine-tuning for NLP can be understood both mathematically and visually.

3.1 Mathematically: Surgical Fine-tuning for NLP

Extending the notation originally defined by Lee et al. (2022b), a pretrained model can be denoted as $f = f_n \circ ... \circ f_1(x)$ where each layer f_i has parameters θ. Surgical fine-tuning w.r.t. to a subset of $S \subseteq \{1, ..., n\}$ layers can the be defined as solving the following objective:

$$\arg\min_{\theta_i, \forall_{i \in S}} \mathcal{L}_{nlp}(f(\theta_1, ..., \theta_n))$$

All non-surgery parameters ($\theta_i \notin S$) are held fixed during fine-tuning.

A key difference in this work from that of Lee et al. (2022b) is that here, I use the loss function $\mathcal{L}_{nlp}$ to represent a sequence or token-level classification loss (typically cross-entropy); Lee et al. (2022b) use $\hat{\mathcal{L}}_{tgt}$, which represents the loss from the target of domain adaptation. The distinction between token and sequence-level classification in NLP is discussed further in Section 4.

3.2 Visually: Surgical Fine-tuning for NLP

Surgical fine-tuning can also be illustrated visually, as shown in Figure 2.

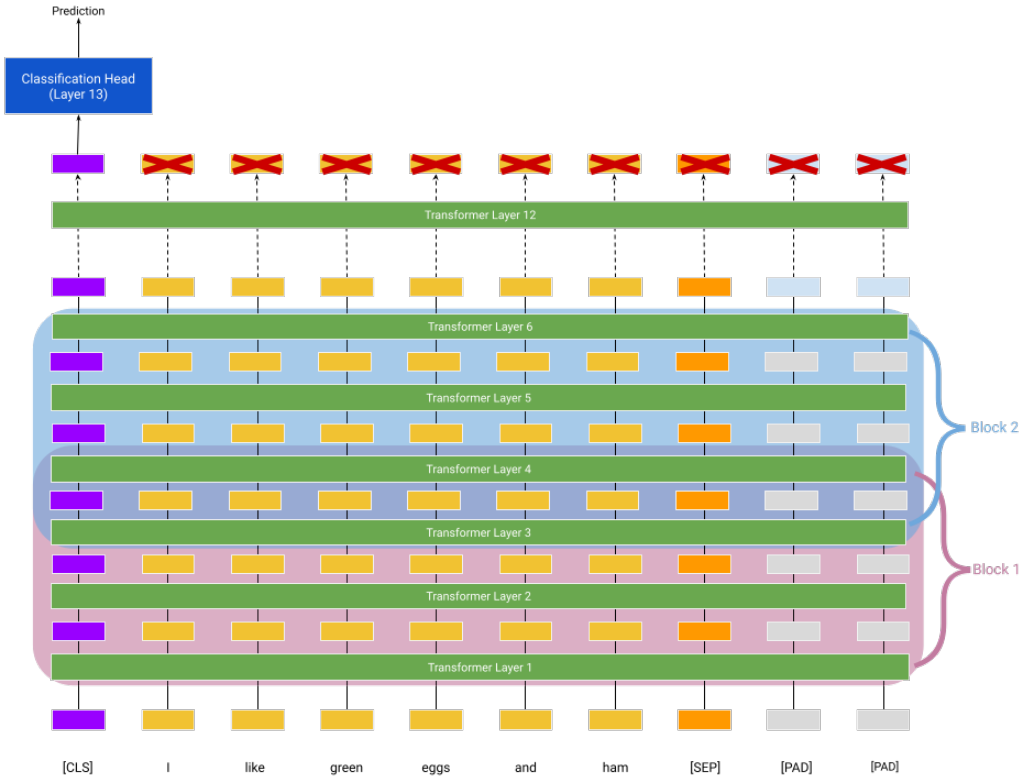


Figure 2: This above illustrates `BertForSequenceClassification`. The final hidden state for `[CLS]` token representation is used to make a prediction and hidden states corresponding to all other tokens are discarded. In surgical fine-tuning, at a time, the parameters of a single block and the classifier head are jointly fine-tuned. For example, in a single round of surgical fine-tuning, we may only update the parameters of layers in Block 1 (pink) as well as the classification head (dark blue). Illustration inspired by (McCormick, 2019).

3.3 Methods

Baselines I tested two baselines: fine-tuning of the last layer only (i.e., the parameters of the classifier head) and fine-tuning the entire model. Although more baselines could be tested, I chose these as they were the baselines also used by (Lee et al., 2022b) and are also commonly used fine-tuning methods in-practice.

Surgical Fine-tuning I tested surgically fine-tuning in contiguous blocks of 4 layers, where each block is separated by a stride of 2. For example, Block 1 consists of layers [1, 4], Block 2 consists of layers [3, 6], etc. In a round of surgical fine-tuning, I fine-tune all of the parameters in the given block, plus the parameters of the classification head. Examples of these blocks are also illustrated in Figure 2.

4 Experiments

In total, I empirically evaluated surgical fine-tuning on four unique NLP tasks. Details of the associated datasets, models, and experiment configurations are also described below.

4.1 Tasks

The tasks I used for experimentation can be divided into two categories: token-level classification and sequence-level classification.

Token-Level Classification (POS, NER, Chunking) In token-level classification, a class prediction is made for each token in the input sequence (sentence). The loss is accumulated over all non-`[PAD]` tokens in the sequence.[1]

The token-level classification datasets I used were from the CoNLL-2003 shared task (Tjong Kim Sang & De Meulder, 2003). In particular, I chose to test surgical fine-tuning on the Part-of-Speech (POS), Named Entity Recognition (NER), and Chunking tasks.

In token-level classification, the input to the classification head is an $n \times h_d$ hidden state matrix, where n is the maximum sequence length and h_d is the hidden state embedding size (n is typically 512 and h_d is typically 768 for BERT). The output is an $n \times m$ matrix, where m is the number of classes.

Sequence-Level Classification (Grammaticality) In sequence-level classification, a single class prediction is made for the entire sequence.

The sequence-level classification dataset I used was from the GLUE shared task (Wang et al., 2018). In particular, I chose to test surgical fine-tuning on the Corpus of Linguistic Acceptability (CoLA), in which sequences are labeled as grammatical or not grammatical (Warstadt et al., 2018). POS, NER, and Chunking could perhaps be thought to rely more heavily on localized, lower-level syntactic information in language, as explained in Section 1; however, to perform well on CoLA, the model must arguably capture holistic and nuanced semantic understanding of the entire sentence. Examples of sequences and labels from this dataset are given in Table 1.

Table 1: Examples sentences and judgements of whether a sentence is grammatical or not, taken from the CoLA dataset. A label of 1 indicates the sentence is grammatical and 0 indicates it is not.

Sentence	Label
"Harry coughed himself into a fit."	1
"Harry coughed himself."	0
"Harry coughed us into a fit."	0

In sequence-level classification, the input to the classification head is an $1 \times h_d$ hidden state embedding corresponding to the `[CLS]` token. [2] The output is a vector of logits of size $1 \times m$, where m is the number of classes.

4.2 Experimental details

Datasets and Metrics The CoNLL-2003 dataset (and each associated task – POS, NER, and Chunking) has 14,041 train and 3,250 validation examples. I fine-tuned using all of the training examples and report the accuracy for each task on the full validation set. I define token-level accuracy by dividing the total number of non-`[PAD]` tokens with correct class predictions over the the total number of non-`[PAD]` tokens. From the CoLA dataset, I used the 8,551 examples from `raw/in_domain_train.tsv` and partitioned it such that a random 90% split was used for training and the remaining 10% set was used for validation. For CoLA, I also report accuracy on the validation set but at the sequence-level.[3]

Models For token-level classification, I used the pretrained `BertForTokenClassification` model available on HuggingFace. Similarly, for sequence-level classification, I used the pretrained `BertForSequenceClassification` model. Both models were initialized using `bert-base-uncased`.

[1] `[PAD]` tokens are special tokens that may be added to the end of a sequence in order to standardize the length of all sequences in the batch.

[2] The `[CLS]` token marks the beginning of a sequence. The hidden state corresponding to this token is typically used for classification.

[3] I chose to report the accuracy metric on CoLA for ease-of-implementation and consistency of reporting with CoNLL-2003, but generally, the authors recommend reporting Matthew's Correlation Coefficient (MCC) for this dataset due to class imbalances (Warstadt et al., 2018).

Training Procedure For each fine-tuning procedure performed on the CoNLL tasks, I fine-tuned for one epoch with a batch size of 8. On the CoLA task, I fine-tuned for four epochs with a batch size of 32. For all methods, I used an initial learning rate of `2e-5` (updated during training by a linear learning rate scheduler) and the Adam optimizer with epsilon `1e-8`, following the guidelines recommended by the authors of BERT (Devlin et al., 2019). Due to limited time and compute, I did not run hyperparameter sweeps. The `BertForTokenClassification` model was slower and more memory-intensive to fine-tune than the `BertForSequenceClassification` model, hence the lower batch size and number of epochs.

4.3 Results

Results from the experiments are shown in Table 2. Each model (represented as cell in the table) took 35-40 minutes to fine-tune on a single Tesla M60 GPU.

In all of the experiments, I did not fine-tune the word or positional embeddings.

Table 2: Accuracy of surgically fine-tuning a 12-layer BERT model plus classification head on a spectrum of NLP tasks. The best performing surgically fine-tuned models per task are in **bold**.

Task	POS	NER	Chunking	Grammaticality
Baselines				
Classifier Head Only	0.378	0.681	0.433	0.71
All: Layers [1, 12]	0.808	0.868	0.834	0.84
Surgical Fine-tuning				
Block 1: Layers [1, 4]	0.783	0.841	0.815	0.78
Block 2: Layers [3, 6]	0.791	0.859	0.822	0.83
Block 3: Layers [5, 8]	**0.791**	**0.864**	**0.825**	**0.84**
Block 4: Layers [7, 10]	0.788	0.861	0.820	0.83
Block 5: Layers [9, 12]	0.787	0.856	0.802	0.80

5 Analysis

Equipped with the results from Table 2, we can attempt to answer the questions posed in Section 1.

5.1 Does surgical fine-tuning improve performance on NLP tasks?

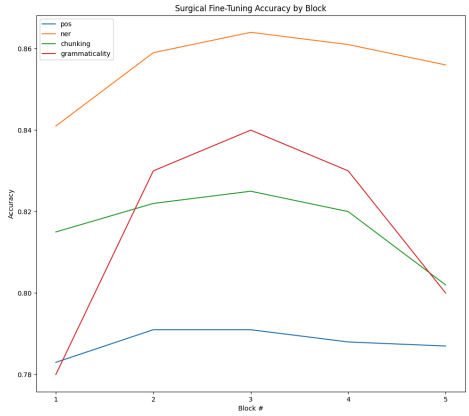
Surgical fine-tuning provided significant benefits on all the NLP tasks that were tested (POS, NER, Chunking, and Grammaticality) compared to fine-tuning the classifier head only (i.e., standard last layer fine-tuning), with the middle layers (Block 3) consistently showing the best performance. While most of the performance gain seems to be attributable to fine-tuning more layers (and therefore parameters) in total, as shown in Figure 3b, the location of the fine-tuned parameters does play a role as well, as shown in Figure 3a.

In the context of these NLP tasks, surgical fine-tuning did generally do slightly worse than fine-tuning all layers, but the gap was relatively small. This result differs from that of Lee et al. (2022b), who found that surgical fine-tuning could sometimes outperform fine-tuning all layers.

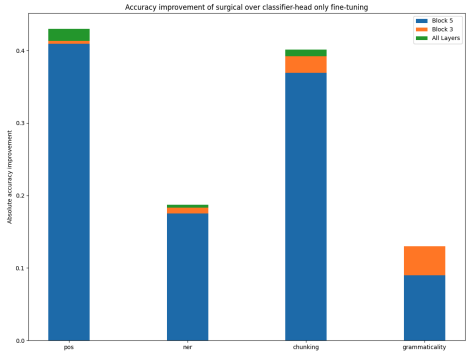
5.2 Does surgical fine-tuning performance reflect the classical NLP pipeline?

The results from these experiments do not appear to reflect the classical NLP pipeline. Rather than achieving peak performance at earlier blocks for syntactic tasks like POS and shifting to later blocks for semantic tasks like Grammaticality, the experiment results revealed a surprisingly consistent trend. Surgical fine-tuning always peaked in performance at the middle block (Block 3). This can be clearly seen in Figure 3a.

One possible explanation for this result is that I did not test surgical fine-tuning on tasks that would reveal the pipeline trend. In their work studying what BERT learns about the structure of language, Jawahar et al. (2019) apply performance-based probing on SentEval (Conneau & Kiela, 2018), which



(a) We can see that surgical fine-tuning performance always peaks at the middle layers (Block 3) for all tasks that were tested.



(b) Absolute accuracy improvements are shown over last-layer (classifier-head only) fine-tuning. We can see that most of the performance improvement comes from fine-tuning more parameters, with the block location (Block 3 vs. Block 5) adding a marginal additional improvement.

Figure 3: These plots illustrate the results of surgical fine-tuning from Table 2.

contains 10 probing tasks that they group into 3 linguistic levels: surface, syntactic, and semantic tasks. They make a similar conclusion to Tenney et al. (2019), i.e., that BERT learns a hierarchy of linguistic features. However, the best performing layers for both syntactic and semantic tasks are between layers 6 and 9, as illustrated in Figure 4. Only what they call surface tasks, particularly sentence length (SL) and word content (SL) from SentEval, perform best at earlier layers (particularly, layers 3 and 4).

My results seem to be harmonious with this work; for all four tasks I performed (POS, NER, Chunking, Grammaticality), which can arguably be classified as syntactic and semantic tasks, Block 3 performed the best. Block 3 (layers [5, 8]) and Block 4 (layers [7, 10]) share the most overlap with layers 6-9. I did not test surgical fine-tuning on any surface task similar to those used by Jawahar et al. (2019), which could explain why I did not see the pipeline trend appear.

Another possible explanation is that the classical NLP pipeline does not quite exist, or rather, there is a more nuanced explanation of BERT's inner workings than layer depth alone. In recent work re-examining whether BERT does, indeed, rediscover a classical NLP pipeline, Niu et al. (2022) question the performance-based probing methods used by Jawahar et al. (2019); Tenney et al. (2019). They suggest their own, attention-based probing mechanism called *GridLoc* instead, and argue that this uncovers a different sort of linguistic structure in BERT. While they do agree that surface tasks attend to lower layers, Niu et al. (2022) argue that syntactic and semantic tasks are inseparable. Furthermore, they show that for most tasks, BERT attends heavily to a single token position, across all layers, as shown in Figure 5

Further research probing which model parameters BERT actually attends to when performing a given NLP task could inform better surgical fine-tuning strategies, as it could allow us to target those specific parameters during fine-tuning (and ultimately, learn to learn which parameters are best to fine-tune).

6 Conclusions

Takeaways In this work, I empirically evaluated surgical fine-tuning, a new technique which has previously been shown to improve domain adaptation of pretrained vision models, but instead using BERT on four standard NLP tasks: three token-level classification tasks (POS, NER, Chunking) and one sequence-level classification task (Grammaticality). I found that surgical fine-tuning *does* improve performance over standard last layer or last few layers fine-tuning. Interestingly, the location of the best block to fine-tune consistently appears in the middle of the model. Thus, the choice of the best block to fine-tune does not appear to reflect the classical NLP pipeline as I had originally hypothesized, but ideally, further experiments would be done to better understand this trend.

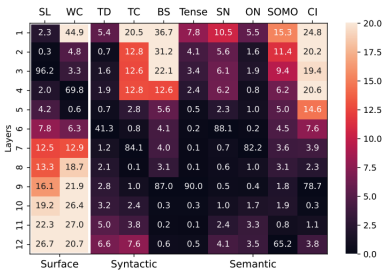
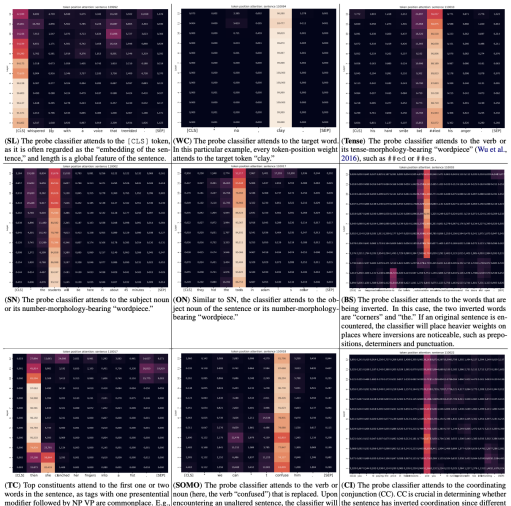


Figure 1: Layer performance probing result of Jawahar et al. (2019), as presented in their Table 2. For clearer visualisation, we transformed this table into a heat map. Each column corresponds to a task, with the best-performing layer in that column containing the performance as a percentage, and the remaining cells displaying their deviation from the best performer in raw percentage points. Surface tasks perform better near the top; syntactic tasks and semantic tasks perform better near the bottom, but their performance patterns are not distinguishable by layer.

Figure 4: Jawahar et al. (2019)'s results newly presented by Niu et al. (2022). Earlier layers are displayed toward the top of the heatmap and later layers are displayed toward the bottom of the heatmap. Niu et al. (2022) conclude that while surface-level tasks do appear to perform better in earlier layers, syntactic and semantic-level tasks perform best between layers 6-9.



(**SL**) The probe classifier attends to the `[CLS]` token, as it is often regarded as the "embedding of the sentence," and length is a global feature of the sentence.

(**WC**) The probe classifier attends to the target word. In this particular example, every token-position weight attends to the target token "clay."

(**Tense**) The probe classifier attends to the verb or its tense-morphology-bearing "wordpiece" (Wu et al., 2016), such as `##ed` or `##s`.

(**SN**) The probe classifier attends to the subject noun or its number-morphology-bearing "wordpiece."

(**ON**) Similar to SN, the classifier attends to the object noun of the sentence or its number-morphology-bearing "wordpiece."

(**BS**) The probe classifier attends to the words that are being inverted. In this case, the two inverted words are "corners" and "the." If an original sentence is encountered, the classifier will place heavier weights on places where inversions are noticeable, such as prepositions, determiners and punctuation.

(**TC**) Top constituents attend to the first one or two words in the sentence, as tags with one presentential modifier followed by NP VP are commonplace. E.g., in this case, the sequence is labelled as `RB_NP_VP_.`

(**SOMO**) The probe classifier attends to the verb or noun (here, the verb "confused") that is replaced. Upon encountering an unaltered sentence, the classifier will attend to common verbs and nouns that are likely to be replaced.

(**CI**) The probe classifier attends to the coordinating conjunction (CC). CC is crucial in determining whether the sentence has inverted coordination since different CCs serve different purposes when connecting two parts of a sentence.

Figure 8: Example token-position attention plots and their pattern analyses. Similar to Figure 7, the attention weight is displayed in a 2-dimensional heat map, with larger weights associated with brighter colours. The tokens of the example sentence are displayed along the x axis. The number on each cell is the attention weight as a percentage. Since attention weight is normalised by softmax at each layer, numbers in every row should sum up to 100.

Figure 5: Niu et al. (2022) show that "for most sentences, the token-position attention at every layer attends to the same token, hence the bright vertical line."

Future Work It would be very worthwhile to test surgical fine-tuning on even more tasks, such as surface-level tasks like SL and WC from SentEval (Conneau & Kiela, 2018) or semantic understanding tasks, like semantic relations or coreference used by Tenney et al. (2019) or the Stanford Sentiment Treebank or Question NLI from GLUE (Wang et al., 2018). This would provide a clearer picture of whether the trend of the middle block giving the best performance remains. More broadly, it would also be interesting to compare surgical fine-tuning against other new fine-tuning techniques that have been proposed in NLP, such as adapters or LoRA (Houlsby et al., 2019; Hu et al., 2021), or to combine it with adaptive/behavioral fine-tuning.

References

Aghajanyan, A., Gupta, A., Shrivastava, A., Chen, X., Zettlemoyer, L., and Gupta, S. Muppet: Massive multi-task representations with pre-finetuning, 2021. URL https://arxiv.org/abs/2101.11038.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T. J., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

Conneau, A. and Kiela, D. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

Croce, F., Andriushchenko, M., Sehwag, V., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. Robustbench: a standardized adversarial robustness benchmark. *CoRR*, abs/2010.09670, 2020. URL https://arxiv.org/abs/2010.09670.

Dai, A. M. and Le, Q. V. Semi-supervised sequence learning. In *NIPS*, 2015.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Han, X. and Eisenstein, J. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4238–4248, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1433. URL https://aclanthology.org/D19-1433.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, 2019.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1031. URL https://aclanthology.org/P18-1031.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., and Chen, W. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685, 2021.

Jawahar, G., Sagot, B., and Seddah, D. What does bert learn about the structure of language? In *Annual Meeting of the Association for Computational Linguistics*, 2019.

Lee, H.-y., Li, S.-W., and Vu, T. Meta learning for natural language processing: A survey. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 666–684, Seattle, United States, July 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.49. URL https://aclanthology.org/2022.naacl-main.49.

Lee, Y., Chen, A. S., Tajwar, F., Kumar, A., Yao, H., Liang, P., and Finn, C. Surgical fine-tuning improves adaptation to distribution shifts, 2022b. URL https://arxiv.org/abs/2210.11466.

Logeswaran, L., Chang, M.-W., Lee, K., Toutanova, K., Devlin, J., and Lee, H. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3449–3460, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1335. URL https://aclanthology.org/P19-1335.

McCormick, C. Bert fine-tuning tutorial with pytorch, Jul 2019. URL https://mccormickml.com/2019/07/22/BERT-fine-tuning/.

Mehri, S., Razumovskaia, E., Zhao, T., and Eskenazi, M. Pretraining methods for dialog context representation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3836–3845, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1373. URL https://aclanthology.org/P19-1373.

Niu, J., Lu, W., and Penn, G. Does bert rediscover a classical nlp pipeline? In *International Conference on Computational Linguistics*, 2022.

Phang, J., Févry, T., and Bowman, S. R. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks, 2018. URL https://arxiv.org/abs/1811.01088.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Rebuffi, S.-A., Bilen, H., and Vedaldi, A. Learning multiple visual domains with residual adapters. In *NIPS*, 2017.

Ruder, S. Recent Advances in Language Model Fine-tuning. http://ruder.io/recent-advances-lm-fine-tuning, 2021.

Schick, T. and Schütze, H. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Conference of the European Chapter of the Association for Computational Linguistics*, 2020.

Stickland, A. C. and Murray, I. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, 2019.

Tenney, I., Das, D., and Pavlick, E. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL https://aclanthology.org/P19-1452.

Tjong Kim Sang, E. F. and De Meulder, F. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 142–147, 2003. URL https://www.aclweb.org/anthology/W03-0419.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461, 2018. URL http://arxiv.org/abs/1804.07461.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments, 2018. URL https://arxiv.org/abs/1805.12471.