Supplementary Material for Many Faces of Feature Importance: Comparing Built-in and Post-hoc Feature Importance in Text Classification

Vivian Lai and **Zheng Cai** and **Chenhao Tan**
Department of Computer Science
University of Colorado Boulder
Boulder, CO
vivian.lai, jon.z.cai, chenhao.tan@colorado.edu

A Preprocessing and Computational Details

Preprocessing. We used spaCy for tokenization and part-of-speech tagging. All the words are lowercased. Table 1 shows basic data statistics.

dataset	average tokens
Yelp	134.6
SST	20.0
Deception	163.7

Table 1: Dataset statistics.

Hyperparameter tuning. Hyperparameters for both SVM and XGBoost are tuned using cross validation. The only hyperparameter tuned for SVM includes C. We try a range of Cs from log space -5 to 5. The finalized value of C ranges between 1 and 5. Hyperparameters tuned for XGBoost include learning rate, max depth of tree, gamma, number of estimators and colsample by tree. We lay out the range of values tried in the process of hyperparameter tuning, learning rate: 0.1 to 0.0001, max depth of tree: 3 to 7, gamma: 1 to 10, number of estimators: 1000 to 10000 and colsample by tree: 0.1 to 1.0. Hyperparameters for LSTM with attention are tuned using the validation dataset which comprises 10% of the entire dataset. They include embedding dimension, hidden dimension, learning rate, number of epochs and the type of optimizer. The range of values tried in the process of hyperparameter tuning, hidden dimension: 256 and 512, learning rate: 0.01 to 0.0001, number of epochs: 3 to 20 and type of optimizer: SGD and adam.

BERT fine-tuning. We fine-tuned BERT from a pre-trained BERT model provided by its original release and pytorch implementation (Wolf, 2019). We use the same architecture of 8 layers Transformer with 12 attention heads. The hidden dimension of each layer is 768. The vocabulary size is 30522. The initial learning rate we use is $5*e^{-5}$, and we add an extra ℓ_2 regularization on the parameters that are not bias terms or normalization layer with a coefficient of 0.01. We do early stopping according to the validation set within the first 20 epochs with batch size no larger than 4. The attention weights we consider are the self-attention weights of the last token of each text instance, namely the attention weights from "[SEP]", since according to BERT's design, the last token will generate the sentence representation fed into the classification layer. For the three target tasks, we choose different maximum lengths according to their natural length. For the deception detection task, the maximum sequence length is 300 tokens. For the SST binary classification task, we choose the default 128 tokens as the maximum length and for the yelp review classification task we use 512 tokens.

BERT alignment. Given that BERT tokenizes a text instance with its own tokenizer, we map the important features from BERT tokens to tokenize results from spaCy we used for other models. To be specific, we generate token start-end information as a tuple and call it token spans. We show an example for text instance "It's a good day.":

tokenization 1: [It's], [a], [good], [day], [.]
token spans 1: (0,3),(4,4),(5,8),(9,11),(12,12)
tokenization 2: [It], ['s], [a], [go], [od], [day], [.]
token spans 2: (0,1), (2,3), (4,4), (5,6), (7,8), (9,11), (12,12)

With the span information, we can identify how a token in the first tokenization relates to tokens in the second tokenization and then aggregate all the attention values to the sub-parts. Formally,

$$W^{(1)}_{(i,j)} = \sum_{(s,t) \text{ s.t. } t \geq i, s \leq j} \min(1, \tfrac{t-i+1}{t-s+1}, \tfrac{j-s+1}{t-s+1}) W^{(2)}_{(k,p)}.$$

In other words, for partial span overlapping, we

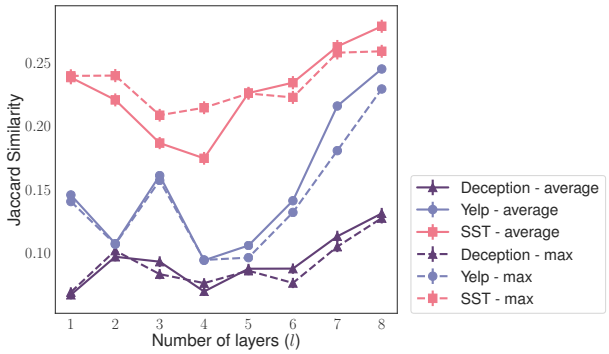


Figure 1: Similarity comparison between BERT layers using average or maximum attention heads score ($k = 10$). In general, similarity becomes greater as l increases. Similarity is higher when average attention heads score is computed.

allocate the weight according to the span overlapping ratio. For example: if $\text{span}_i^{(1)} = (2, 5)$ and $\text{span}_{k-1}^{(2)} = (2, 3), \text{span}_k^{(2)} = (4, 6)$, then $W_{(2,5)}^{(1)} = W_{(2,3)}^{(2)} + \frac{2}{3}W_{(4,6)}^{(2)}$. Here $W^{(2)}$ represents the importance weight according to the second tokenization, $W_{(i,j)}^{(1)}$ represents the aligned feature importance for the token that has span (i, j) in the first tokenization. By definition, $\sum_{(i,j)} W_{(i,j)}^{(1)} = \sum_{(i,j)} W_{(i,j)}^{(2)} = 1$ for attention values.

LIME. We use the LimeTextExplainer and write a wrapper function that returns actual probabilities of the respective model. Since the LinearSVM generates only binary predictions, we return 0.999 and 0.001 instead. We use 1,000 samples for fitting the local classifier.

SHAP. We use a LinearExplainer for linear SVM, a TreeExplainer for XGBoost, and adapt the gradient-based DeepExplainer for our neural models. The main adaptation required for the neural method is to view the embedding lookup layer as a matrix multiplication layer so that the entire network is differentiable on the input token ids.

B Additional Figures

Similarity between BERT layers and SVM (ℓ_2). The final layer is more similar than other layers. See Figure 1.

Built-in similarity is much lower with deep learning models, and post-hoc methods "smooth" the distance. Similar results are observed in SVM (ℓ_1) and BERT. See Figure 2.

Similarity between methods is lower for deep learning models. Similar results are observed in SVM (ℓ_1), XGBoost and BERT. See Figure 3.

Similarity vs. predicted labels. Similarity is not necessarily higher when predictions agree, it is also not necessarily lower when predictions disagree. See Figure 4 and Figure 5.

Similarity vs. length. The negative correlation between length and similarity grows stronger as k grows. See Figure 6.

Similarity vs. type-token ratio. The positive correlation between type-token ratio and similarity grows stronger as k grows. See Figure 7 and Figure 8.

Entropy. Deep learning models generate more diverse important features than traditional models. See Figure 9.

Jensen-shannon distance between POS. Distance of part-of-speech tag distributions between important features and all words is generally smaller with post-hoc methods for traditional models. See Figure 10.

References

Thomas Wolf. 2019. huggingface/pytorch-pretrained-bert. https://github.com/huggingface/pytorch-pretrained-BERT.

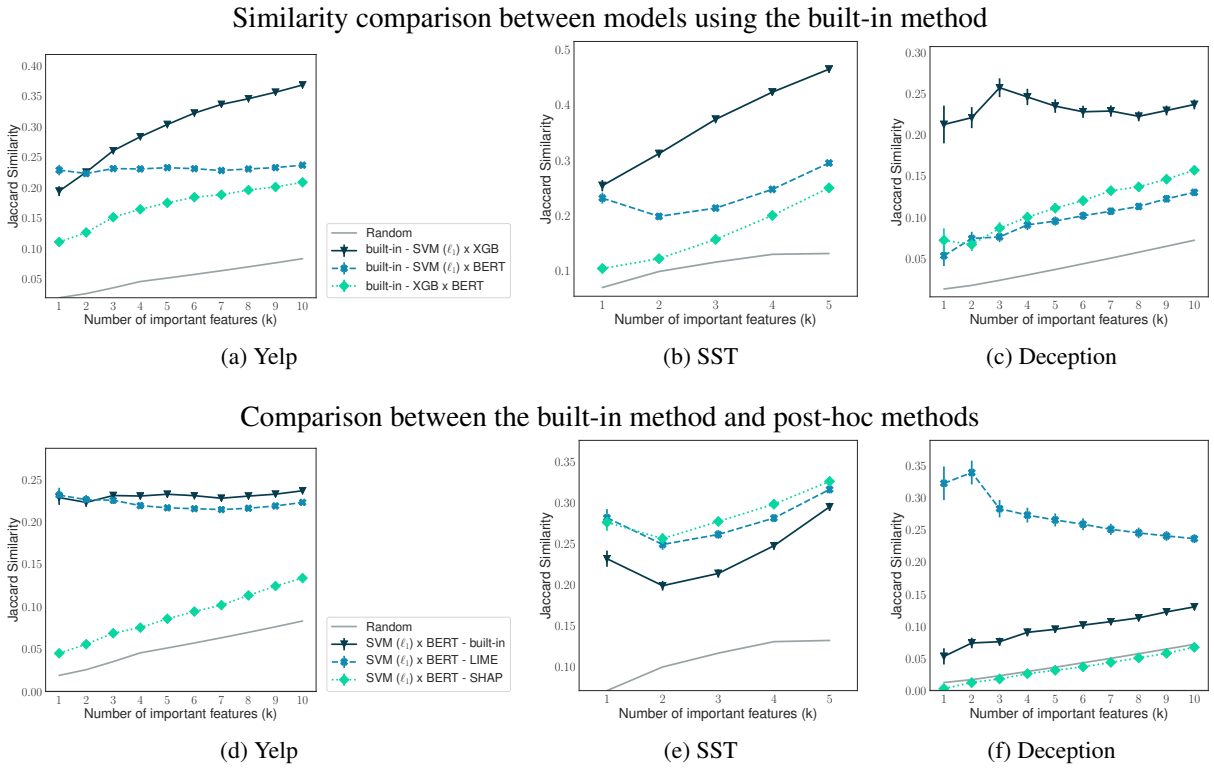


Figure 2: Similarity comparison between models with the same method. x-axis represents the number of important features that we consider, while y-axis shows the Jaccard similarity. *Error bars represent standard error throughout the paper.* The top row compares three pairs of models using the built-in method, while the second row compares three methods on SVM (ℓ_1) and BERT. The random line is derived using the average similarity between two random samples of k features from 100 draws.

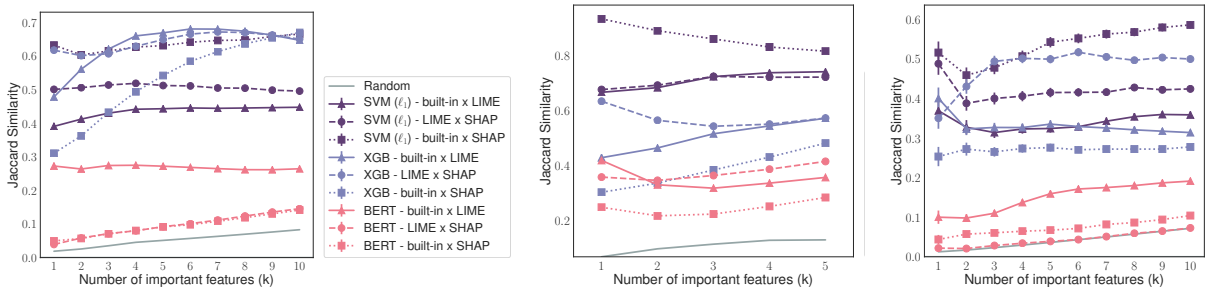
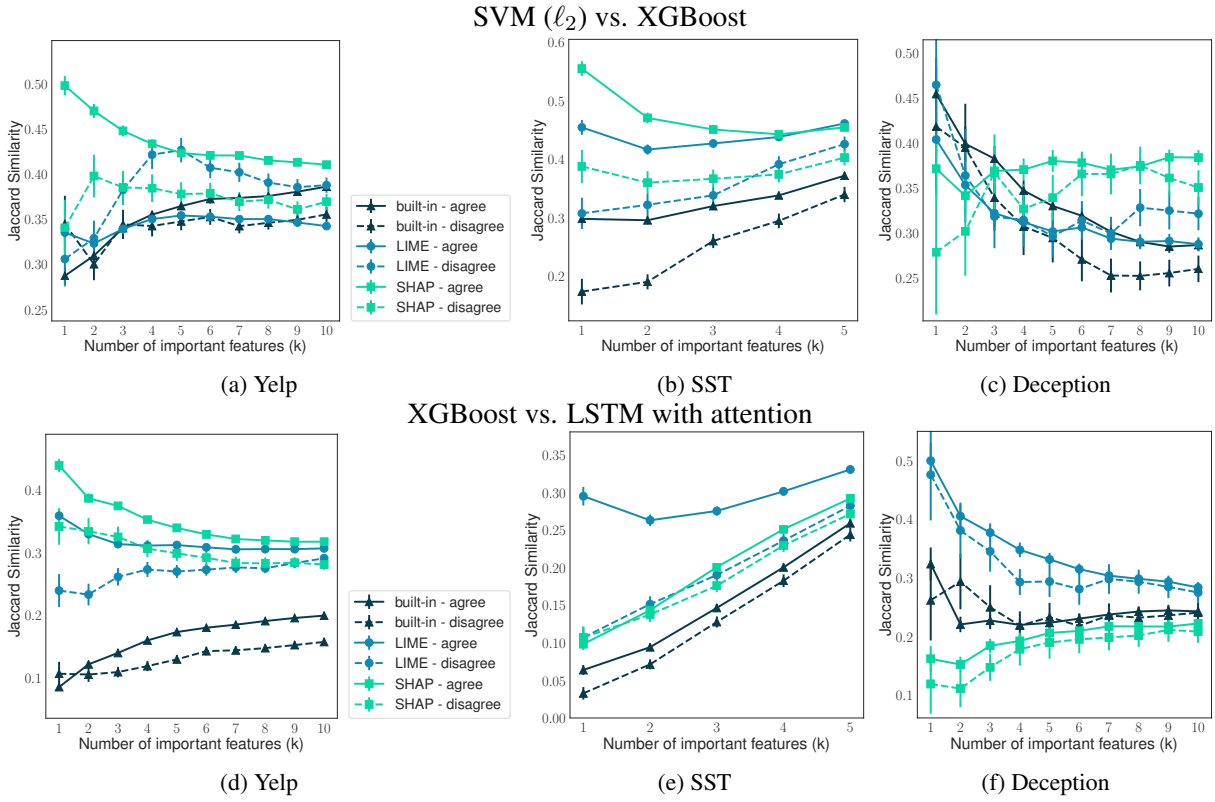


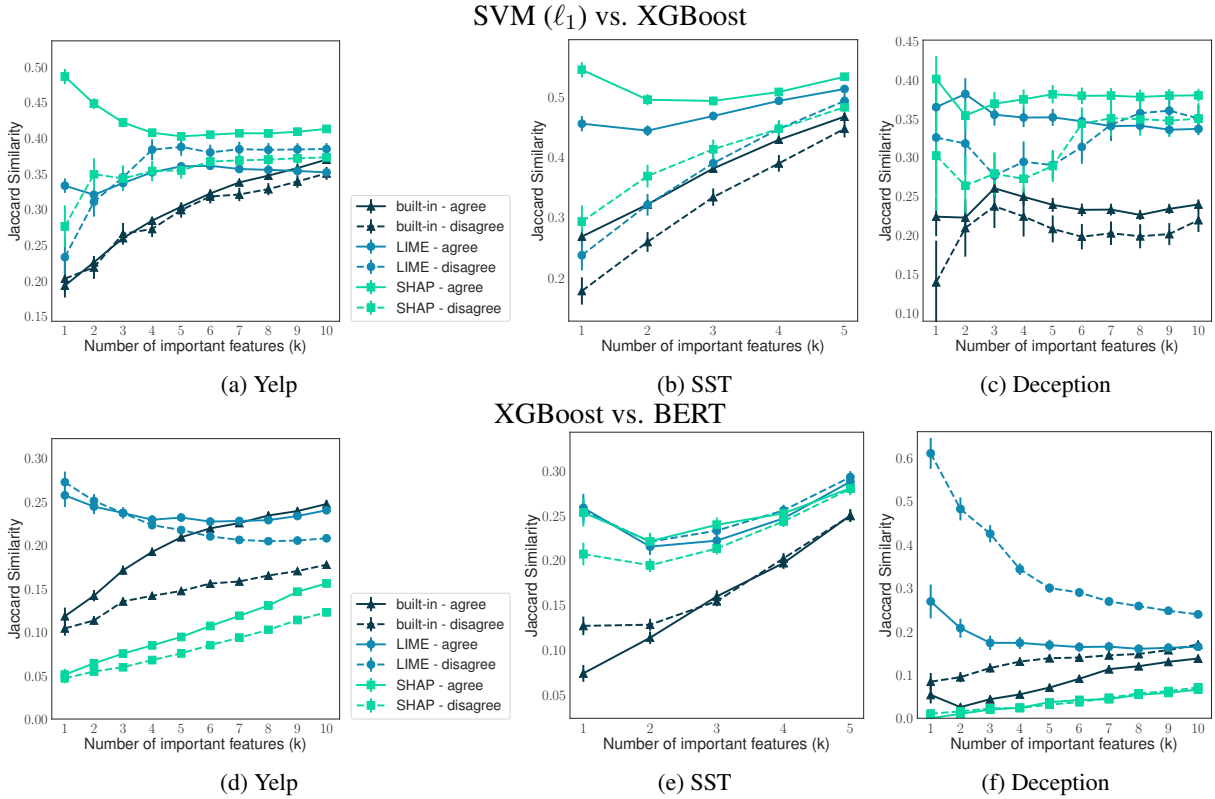
Figure 3: Similarity comparison between methods using the same model for SVM (ℓ_1), XGBoost, and BERT. BERT is much closer to random in deception.



(a) Yelp (b) SST (c) Deception

(d) Yelp (e) SST (f) Deception

Figure 4: Similarity between two models is not necessarily greater when they agree on the predictions, and sometimes, e.g., SVM (ℓ_2) x XGB with LIME method, it is sometimes lower than when they disagree on the predicted labels.



(a) Yelp (b) SST (c) Deception

(d) Yelp (e) SST (f) Deception

Figure 5: Similarity between two models is not necessarily greater when they agree on the predictions, and in some scenarios, e.g., SVM (ℓ_1) x XGB with LIME method, XGB x BERT with LIME method, and XGB x BERT with built-in method, they are sometimes lower than when they disagree on the predicted labels.

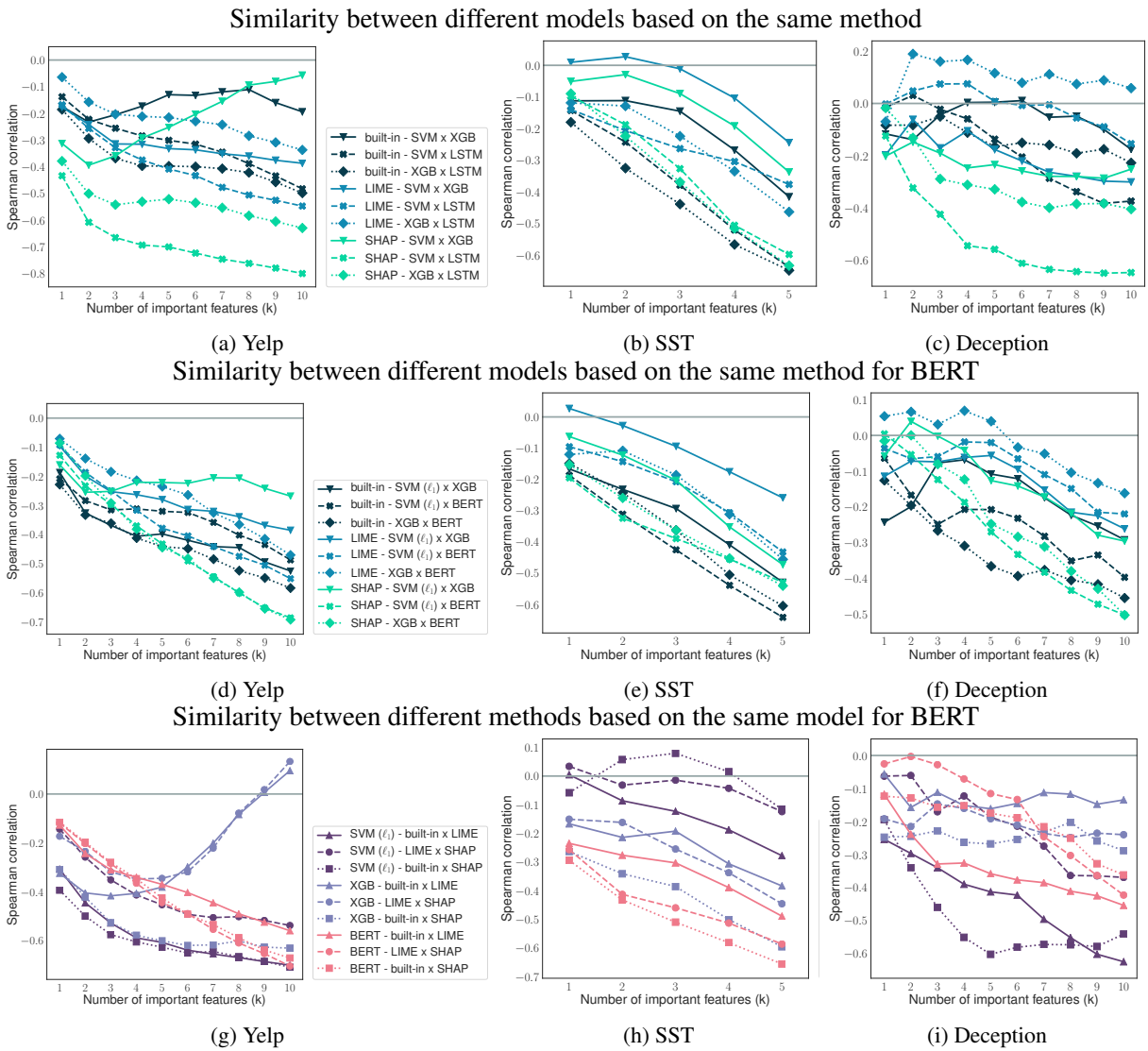
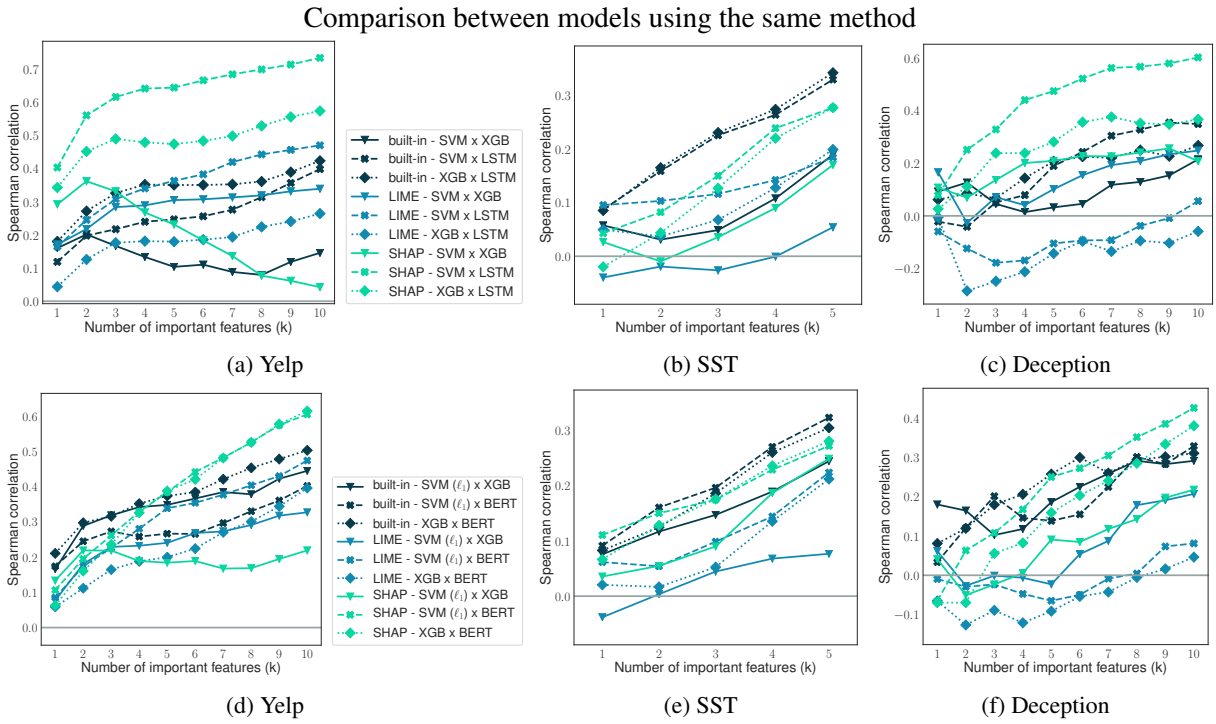


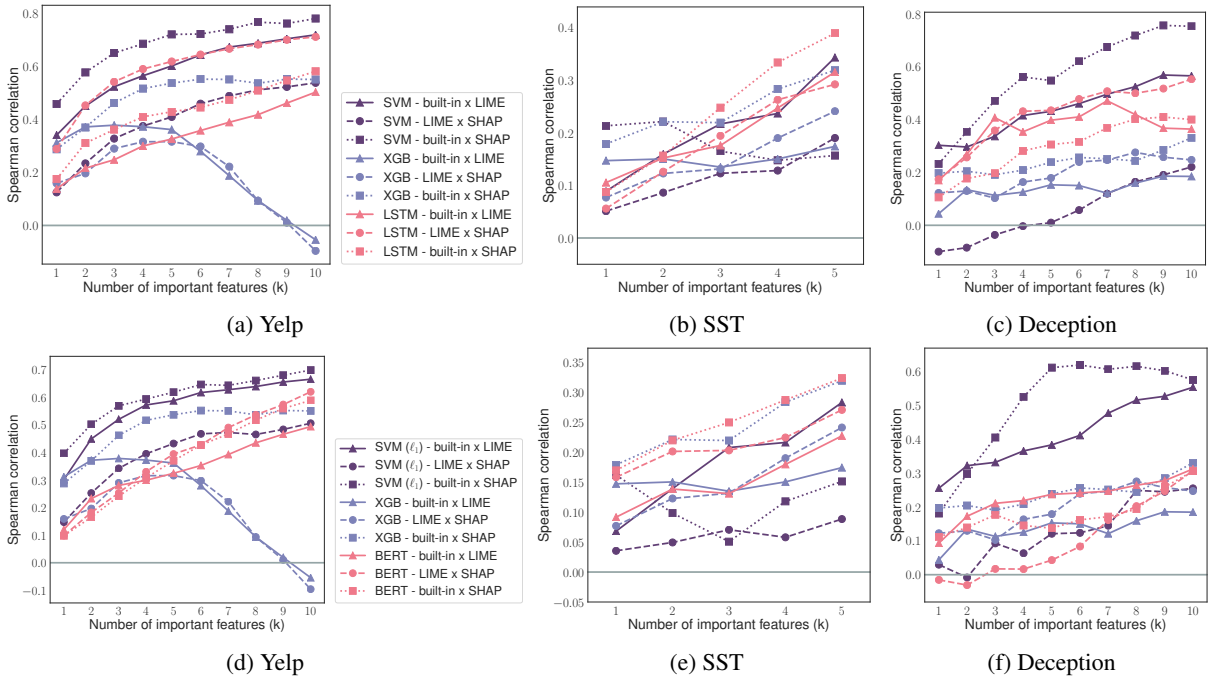
Figure 6: Similarity comparison vs. length. The longer the length of an instance, the less similar the important features are. The negative correlation becomes stronger as k grows. In certain scenarios, e.g., XGB - built-in x LIME and XGB - LIME x SHAP, correlation occasionally goes above 0.



(a) Yelp (b) SST (c) Deception

(d) Yelp (e) SST (f) Deception

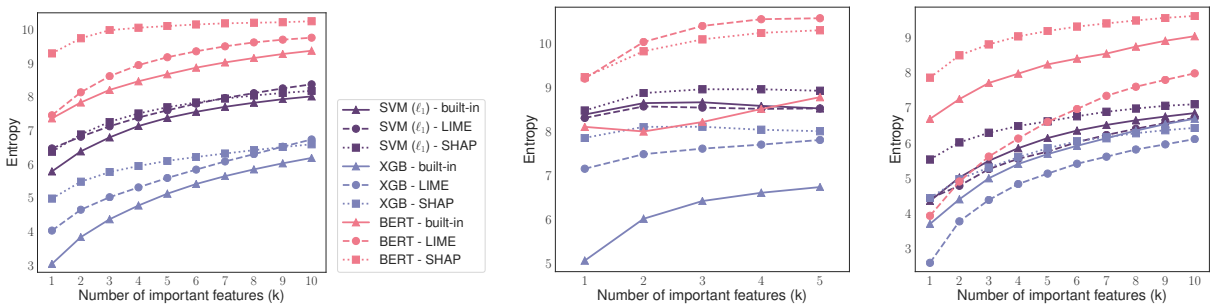
Figure 7: Similarity comparison vs. type-token ratio. The higher the type-token ratio, the more similar the important features are. The positive correlation becomes stronger as k grows. In some cases, e.g., LIME method on deception dataset, correlation becomes weaker as k grows.



(a) Yelp (b) SST (c) Deception

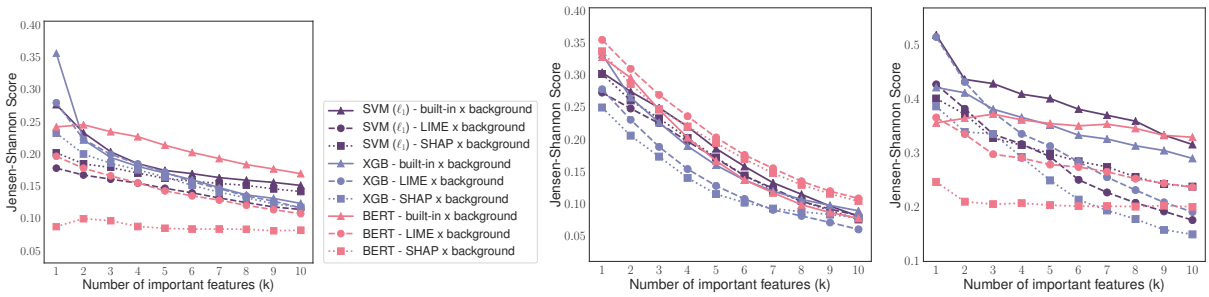
(d) Yelp (e) SST (f) Deception

Figure 8: Similarity comparison vs. type-token ratio. The higher the type-token ratio, the more similar the important features are. The positive correlation becomes stronger as k grows. In some cases, e.g., XGB - built-in and LIME and XGB - LIME and SHAP on Yelp dataset, correlation becomes weaker as k grows.



(a) Yelp (b) SST (c) Deception

Figure 9: The entropy of important features. In general, BERT generates more diverse important features than SVM (ℓ_1) and XGBoost.



(a) Yelp (b) SST (c) Deception

Figure 10: Distance of the part-of-speech tag distributions between important features and all words (background). Distance is generally smaller with post-hoc methods for all models, although some exceptions exist for LSTM with attention and BERT.