BERT

Грабовой Андрей Валериевич

Московский физико-технический институт

МФТИ, г. Долгопрудный

Содержание

- Архитектура BERT.
- Головы BERT.
- Как обучается.
- Применение.
- Где же prior?

Архитектура BERT

Разделяется условно на три части:

1. Tokens Embedding.
2. Self-Attention.
3. Pooler.

Пусть задано множество токенов:

$$\mathcal{I} = \{\mathbf{i} | \mathbf{i} = [0, \cdots, 0, 1, 0, \cdots, 0]^\mathsf{T}\}$$

Задано множество предложений и множество типов токенов в предложении:

$$\mathcal{S} = \mathcal{I}^n, \quad \mathcal{T} = \{[0, 1]^\mathsf{T}, [1, 0]^\mathsf{T}\}^n$$

Отображения:

$$BM_1 : \mathbb{R}^{n \times L} \times \mathbb{R}^{2 \times L} \to \mathbb{R}^{n \times l}$$

$$BM_2 : \mathbb{R}^{n \times L} \times \mathbb{R}^{2 \times L} \to \mathbb{R}^{1 \times l}$$

Суперпозиция отображений:

$$BM_1 = BL_m \circ \cdots \circ BL_1 \circ BSE$$

$$BM_2 = BP \circ BL_m \circ \cdots \circ BL_1 \circ BSE$$

Функция BSE:

$$BSE : \mathbb{R}^{n\times L} \times \mathbb{R}^{n\times 2} \to \mathbb{R}^{n\times l}.$$

Для произвольной матрицы $\mathbf{s} \in \mathcal{S} \subset \mathbb{R}^{n\times L}$ и матрицы $\mathbf{t} \in \mathcal{T} \subset \mathbb{R}^{n\times 2}$ отображение BSE принимает следующий вид:

$$BSE\big(\mathbf{s},\mathbf{t}\big) = \frac{\mathbf{h}_{bse} - \mathsf{E}\mathbf{h}_{bse}}{\sqrt{\mathsf{D}\mathbf{h}_{bse} + \varepsilon}} \cdot \mathbf{w}_1 + \mathbf{w}_2, \quad \mathbf{h}_{bse} = \mathbf{s}\mathbf{W}_1 + \mathbf{1}_{n\times n}\mathbf{W}_2 + \mathbf{t}\mathbf{W}_3,$$

где $\mathbf{W}_1 \in \mathbb{R}^{L\times l}$, $\mathbf{W}_2 \in \mathbb{R}^{n\times l}$, $\mathbf{W}_3 \in \mathbb{R}^{2\times l}$.
Функция BSE имеет настриваемые параметры: $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{w}_1, \mathbf{w}_2$.
Результат работы функции BSE обозначим:

$$\mathbf{h}_0 = BSE\big(\mathbf{s},\mathbf{t}\big),$$

где $\mathbf{h} \in \mathbb{R}^{n\times l}$.

Функция BL:

$$BL : \mathbb{R}^{n\times l} \to \mathbb{R}^{n\times l}.$$

Для матрицы $\mathbf{h} \in \mathbb{R}^{n\times l}$ BL принимает следующий вид:

$$BL_q(\mathbf{h}) = \frac{\mathbf{u}\mathbf{W}_{3+6q} + \mathbf{a} - \mathsf{E}\left(\mathbf{u}\mathbf{W}_{3+6q} + \mathbf{a}\right)}{\sqrt{\mathsf{D}\left(\mathbf{u}\mathbf{W}_{3+6q} + \mathbf{a}\right) + \varepsilon}} \cdot \mathbf{w}_{3+4q} + \mathbf{w}_{4+4q},$$

$$\mathbf{u} = \sigma\big(\mathbf{a}\mathbf{W}_{4+6q}\big), \quad \mathbf{a} = \frac{\mathbf{c}\mathbf{W}_{5+6q} - \mathsf{E}\mathbf{c}\mathbf{W}_{5+6q}}{\sqrt{\mathsf{D}\mathbf{c}\mathbf{W}_{5+6q} + \varepsilon}} \cdot \mathbf{w}_{5+4q} + \mathbf{w}_{6+4q}$$

$$\mathbf{c} = [\mathbf{c}_1, \cdots \mathbf{c}_{r_2}]$$

$$\mathbf{c}_j = \operatorname{softmax}\big(\mathbf{h}\mathbf{W}^j_{6+6q} \odot \mathbf{h}\mathbf{W}^j_{7+6q}\big) \odot \mathbf{h}\mathbf{W}^j_{8+6q}$$

где для всех q матрицы $\mathbf{W}^j_{6+6q}, \mathbf{W}^j_{7+6q}, \mathbf{W}^j_{6+6q} \in \mathbb{R}^{l\times r}$, для всех j матрицы $\mathbf{c}_j \in \mathbb{R}^{n\times r}$, для всех q матрицы $\mathbf{W}^j_{5+6q} \in \mathbb{R}^{l\times l}, \mathbf{W}^j_{4+6q} \in \mathbb{R}^{l\times p}, \mathbf{W}^j_{3+6q} \in \mathbb{R}^{p\times l}$, матрица $\mathbf{c}, \mathbf{a} \in \mathbb{R}^{n\times l}$, матрица $\mathbf{u} \in \mathbb{R}^{n\times p}$.
Настраиваемые параметры:
$\mathbf{W}^j_{3+6q}, \mathbf{W}^j_{4+6q}, \mathbf{W}^j_{5+6q}, \mathbf{W}^j_{6+6q}, \mathbf{W}^j_{7+6q}, \mathbf{W}^j_{8+6q}, \mathbf{w}_{3+4q}, \mathbf{w}_{4+4q}, \mathbf{w}_{5+4q}, \mathbf{w}_{6+4q}$
Результат работы функции BL_q:

$$\forall q \in \{1, \cdots m\} \quad \mathbf{h}_q = BL_q\big(\mathbf{h}_{q-1}\big).$$

Функция BP:

$$BP : \mathbb{R}^{n \times l} \to \mathbb{R}^{n \times l}.$$

Для матрицы $\mathbf{h}_m \in \mathbb{R}^{n \times l}$ BP принимает следующий вид:

$$BP\big(\mathbf{h}_m\big) = \sigma\big(\mathbf{h}_m^1 \mathbf{W}_{9+6m}\big),$$

где $\mathbf{h}_m^1$ первая строка матрицы $\mathbf{h}_m$, а матрица $\mathbf{W}_{9+6m} \in \mathbb{R}^{l \times l}$
Функция BP имеет настраиваемые параметры $\mathbf{W}_{9+6m}$
Результат работы функции BP:

$$\mathbf{h} = BP\big(\mathbf{h}_m\big).$$

Вернемся к суперпозициям BM_1, BM_2:

$$\mathbf{h} = BP\big(\mathbf{h}_m\big), \quad \mathbf{h}_q = BM\big(\mathbf{h}_{q-1}\big), \quad \mathbf{h}_0 = BSE\big(\mathbf{s}, \mathbf{t}\big)$$

Получаем вектор эмбедингов слов:

$$BM_1\big(\mathbf{s}, \mathbf{t}\big) = \mathbf{h}_m,$$

Получаем вектор эмбединга предложения:

$$BM_2\big(\mathbf{s}, \mathbf{t}\big) = \mathbf{h}$$

Математическая модель Bert (multi task learning)

LM модель:

$$\mathbf{v} = \text{softmax}\big(\mathbf{h}_m \mathbf{W}_{LM}\big),$$

где $\mathbf{W}_{LM} \in \mathbb{R}^{l \times L}$, а $\mathbf{v}$ это вероятность каждого токена.
NSP модель:

$$z = \sigma\big(\mathbf{h}\mathbf{W}_{NSP}\big),$$

где $\mathbf{W}_{NSP} \in \mathbb{R}^{l \times 1}$, а z это вероятность класса 1.
Функция ошибки:

$$L\big(\mathbf{S}, \mathbf{y}\big) = \sum_{\mathbf{s}_i, \mathbf{t}_i \in \mathbf{S}} CrossEntropy\big(\mathbf{v}_i, \mathbf{s}_i\big) + \sum_{\mathbf{s}_i, \mathbf{t}_i \in \mathbf{S}, y_i \mathbf{y}} CrossEntropyLoss\big(z_i, y_i\big)$$

Задача оптимизации:

$$L\big(\mathbf{S}, \mathbf{y}\big) \to \min_{\mathbf{W}_{all}}$$

Все параметры:

$$\begin{aligned}\mathbf{W}_{all} = [&\mathbf{W}_{LM}, \mathbf{W}_{NSP},\\ &\mathbf{W}_{9+6m},\\ &\mathbf{W}^j_{3+6q}, \mathbf{W}^j_{4+6q}, \mathbf{W}^j_{5+6q}, \mathbf{W}^j_{6+6q}, \mathbf{W}^j_{7+6q}, \mathbf{W}^j_{8+6q}, \mathbf{w}_{3+4q}, \mathbf{w}_{4+4q}, \mathbf{w}_{5+4q}, \mathbf{w}_{6+4q},\\ &\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{w}_1, \mathbf{w}_2]\end{aligned}$$

Tokens Embedding

```
BertEmbeddings(
  (word_embeddings): Embedding(119547, 768, padding_idx=0)
  (position_embeddings): Embedding(512, 768)
  (token_type_embeddings): Embedding(2, 768)
  (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
  (dropout): Dropout(p=0.1, inplace=False)
)
```

```
BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
```

Pooler

```
BertPooler(
  (dense): Linear(in_features=768, out_features=768, bias=True)
  (activation): Tanh()
)
```

Головы BERT

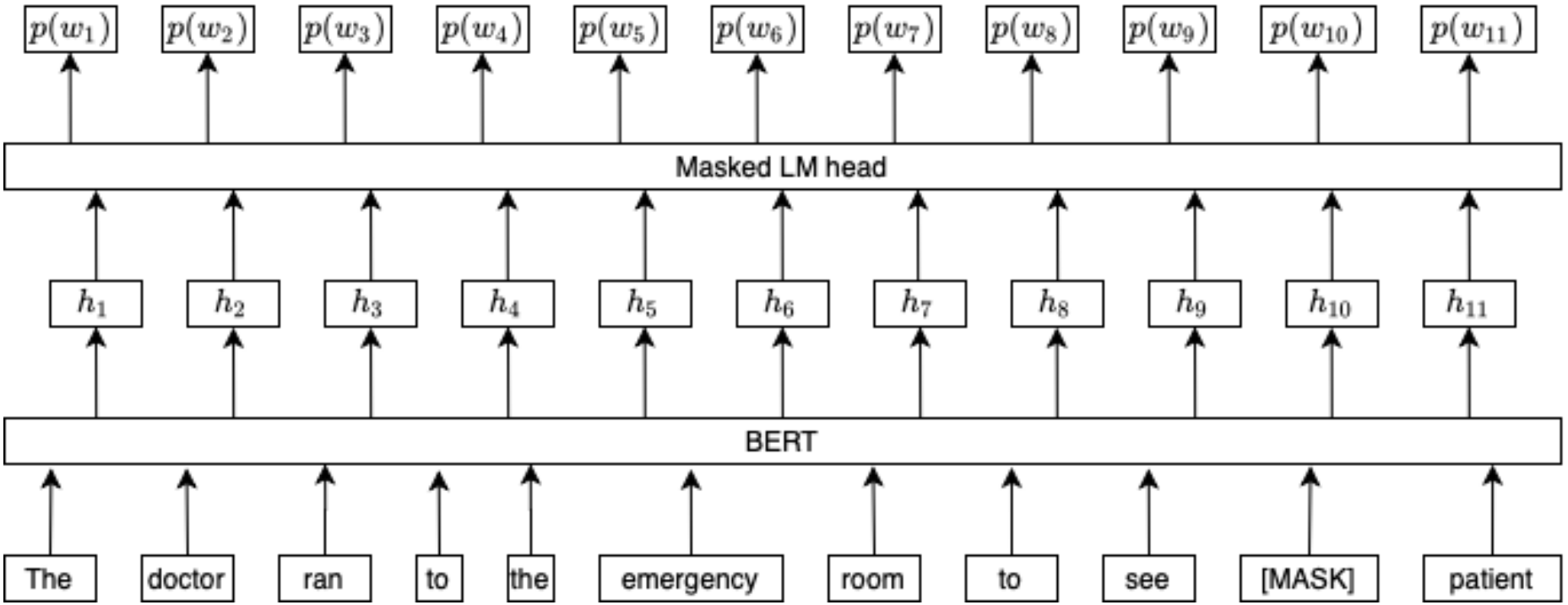
До этого мы разобрали архитектуру генерации признакового описания токенов при помощи модели BERT. Теперь перейдем к использованию сгенерированых признаков для разных задач:

1. Masked LM.
2. Next Sentence Prediction (NSP).

Masked LM

```
BertOnlyMLMHead(
  (predictions): BertLMPredictionHead(
    (transform): BertPredictionHeadTransform(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    )
    (decoder): Linear(in_features=768, out_features=119547, bias=True)
  )
)
```

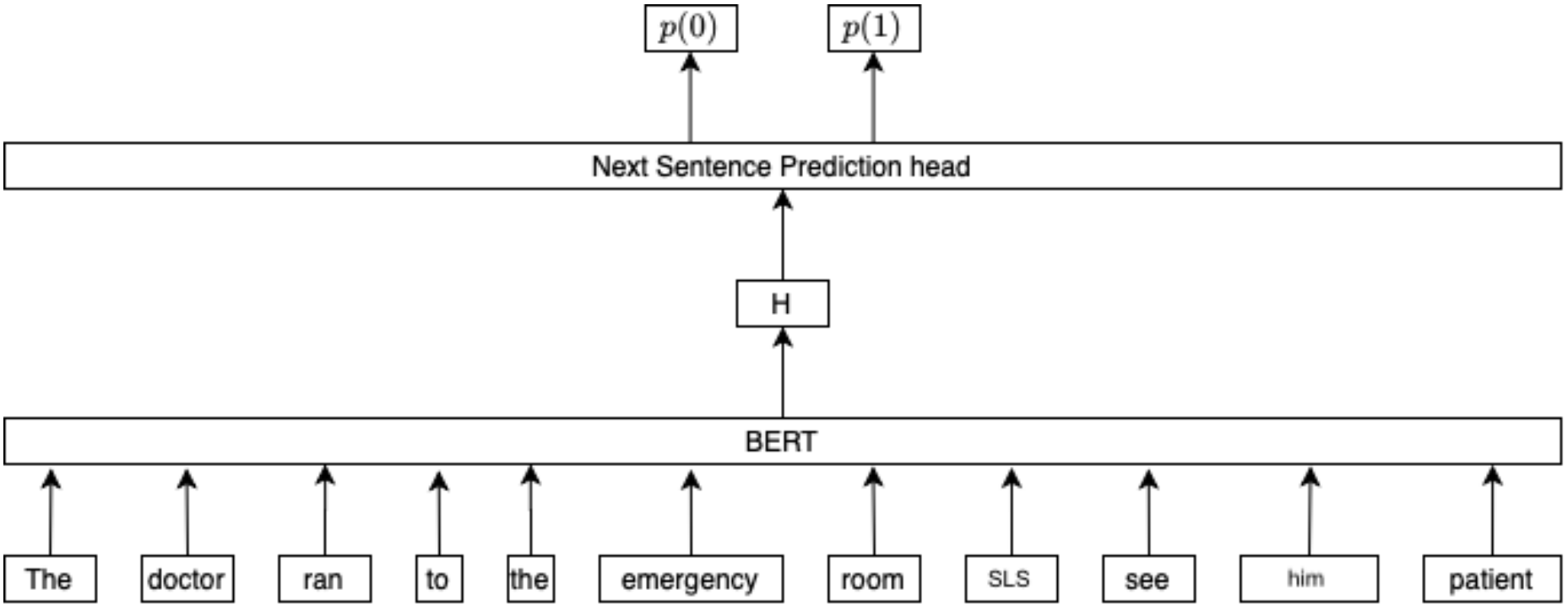
Masked LM обучение



Next Sentence Prediction (NSP)

```
BertOnlyNSPHead(
  (seq_relationship): Linear(in_features=768, out_features=2, bias=True)
)
```

Next Sentence Prediction (NSP) обучение



BERT применение

Обученые параметры модели BERT применяются как генерация признакового описания токенов предложения. После чего применяются простые модели (обычно линейные), которые решают поставленную задачу машиного обучения. Примеры задач:

1. Words in Context.
2. BoolQ.
3. Choice of Plausible Alternatives.
4. и.т.д.

Полезные ссылки на датасеты:

1. https://super.gluebenchmark.com/tasks
2. https://russiansuperglue.com

Где же prior?

1. Основная идея: смысл текста не зависит от решаемой задачи.
2. Сама модель BERT посути является некоторым априорным заданиям векторов, которые потом подстраиваются под конкретную задачу.

Литература

1. J. Devlin, M. Chang, K. Lee, K. Toutanova. BERT: Pre-training of Deep Bidirectiona Transformers for Language Understanding. 2018.
2. J. Ba, J. Kiros, G. Hinton Layer Normalization. 2016.
3. `https://huggingface.co/transformers/model_doc/bert.html#bertmodel`