به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین پنجم

نام و نام خانوادگی	سارا رستمی – امین شاهچراغی
شماره دانشجویی	۸۱۰۱۰۰۳۵۵ - ۸۱۰۱۹۹۱۹۶
تاریخ ارسال گزارش	۱۴۰۱.۱۰.۱۸

فهرست

شکل‌ها

جدول‌ها

No table of figures entries found.

پاسخ ۱. آشنایی با مفهوم توجه و پیاده‌سازی مدل BERT

۱-۱. پیاده‌سازی کدگذار

۱.

به فرآیندی که در آن بر روی قسمتی از ورودی وزن بیشتر و بر روی قسمتی وزن کمتر اعمال می شود، attention گفته می شود. لایه توجه در مدل BERT سه پارامتر Key, Query و Value را به عنوان ورودی می‌گیرد و یک تبدیل خطی را برای تولید دینامیک وزن‌ها برای connectionهای مختلف آغاز می‌کند. و سپس این تبدیل خطی را به یک ضرب داخلی scale شده می‌دهد.

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (۱)$$

در فرمول (۱)، K نشان‌دهنده‌ی Key، V نشان‌دهنده‌ی Value و Q نشان‌دهنده‌ی Query می‌باشد.

۲.

Multi-head attention این قابلیت را دارد که همزمان به چند پوزیشن متمرکز شود در حالی که می تواند sparsity را نیز حفظ کند. در واقع این نوع توجه این قابلیت را به مدل می‌دهد که دسته‌ی گسترده‌تری از روابط بین کلمات را capture کند(نسبت به single-head).

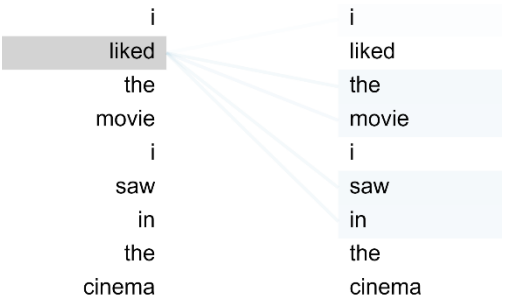
۱-۲. پیاده‌سازی مدل BERT

۱.

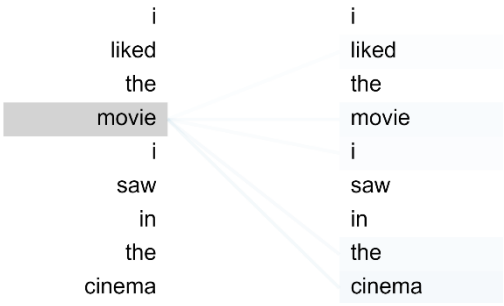
در segment embedding در واقع شماره جمله مورد نظر یا محل واقع شدن آن به مقدار عددی(بردار عددی) تبدیل می شود. به طور مثال در BERT مدل باید بداند که یک token به جمله‌ی A تعلق دارد یا جمله‌ی B. این هدف با تولید یک token ثابت (fixed) به نام segment embedding انجام می‌شود (یک توکن ثابت برای جمله‌ی A و یک توکن ثابت برای جمله‌ی B). پو بردار بازنمایی در لایه‌ی segment embedding وجود دارد. توکن‌های متعلق به ورودی ۱ به اولین بردار assign می‌شوند (index 0) و توکن‌های متعلق به ورودی ۲ به دومین بردار assign می‌شوند (index 1).

۲.

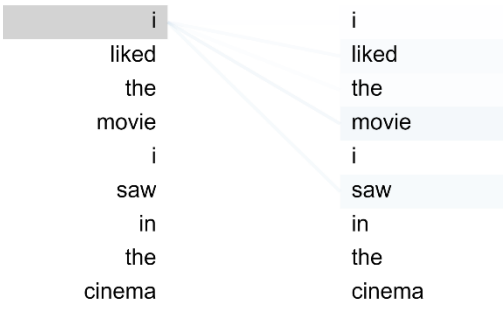
جمله ”I liked the movie I saw in the cinema“ به دسته بند داده شد و مقادیر لایه attention آن استخراج شد. وزن های attention در انتهای کد نمایش داده شده اند. به طور مثال بیشترین وزن کلمه ”movie“ مربوط به کلمات ”I“ و ”cinema“ است. خروجی مدل برای این جمله را در شکل‌های ۱ و ۲ و ۳ مشاهده می‌کنید.



شکل ۱- خروجی مدل **BERT** برای جمله مذکور



شکل ۲- خروجی مدل **BERT** برای جمله مذکور (ادامه)



شکل ۳- خروجی مدل **BERT** برای جمله مذکور(ادامه)

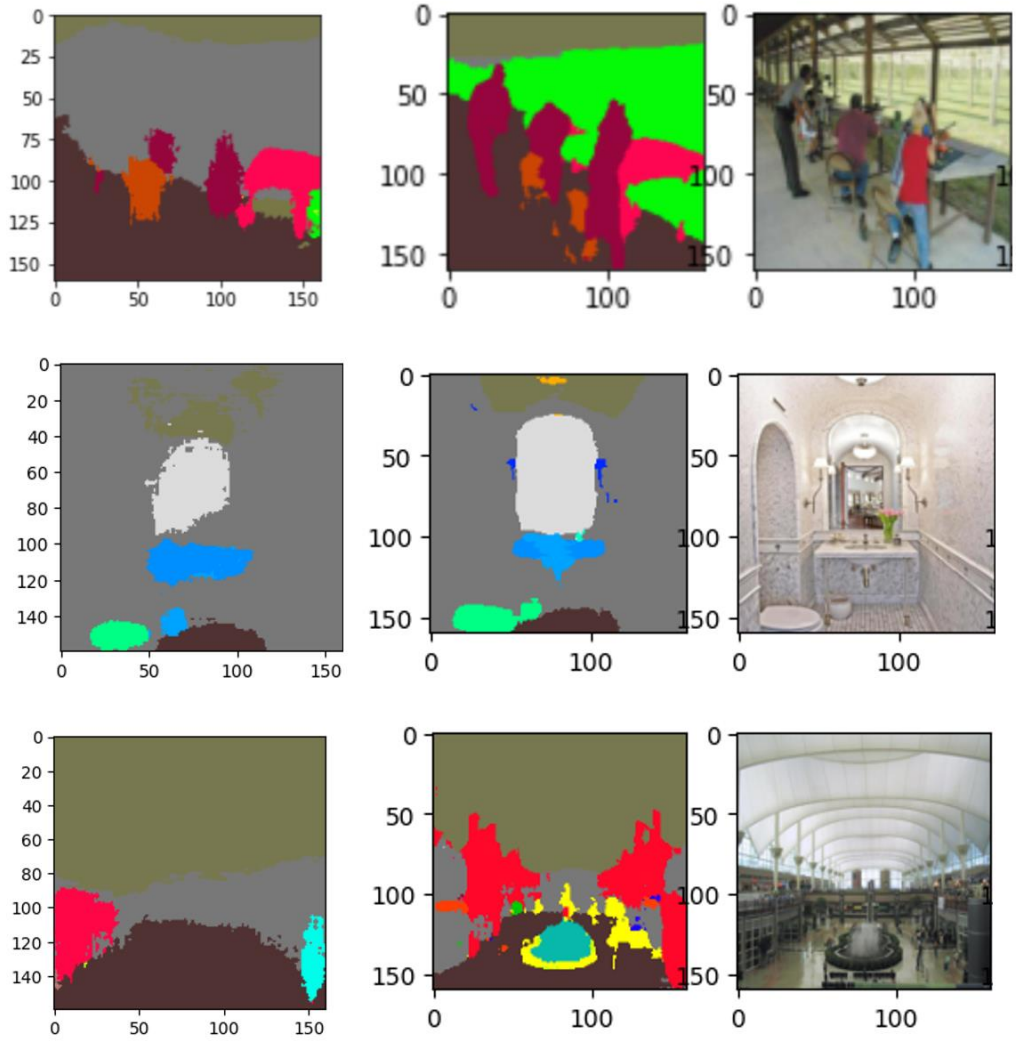
کد های تکمیل شده در نوت‌بوک Q1_transformer_completed.ipynb آمده است .

پاسخ ۲. آشنایی با مفهوم تبدیل‌کننده‌ها در تصویر

۲-۱. آشنایی با مدل BEiT

۲-۲. تقسیم‌بندی معنایی تصویر

در شکل زیر، تصویر ستون سمت راست تصویر اصلی، ستون وسط مربوط به تقسیم‌بندی معنایی موجود در دیتاست و ستون سمت چپ مربوط به تقسیم‌بندی معنایی بدست آمده توسط مدل بازآموزش داده‌ی ما می‌باشد.



شکل ۴- تصاویر سگمنت‌شده پیش‌فرض و مدل بازآموزش‌یافته

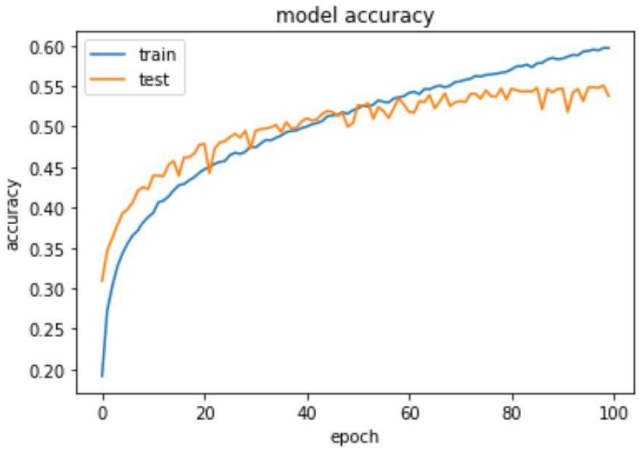
۲-۳. طبقه‌بندی تصاویر

مشخصات مدل MLP طراحی شده را در شکل زیر مشاهده می‌کنید.

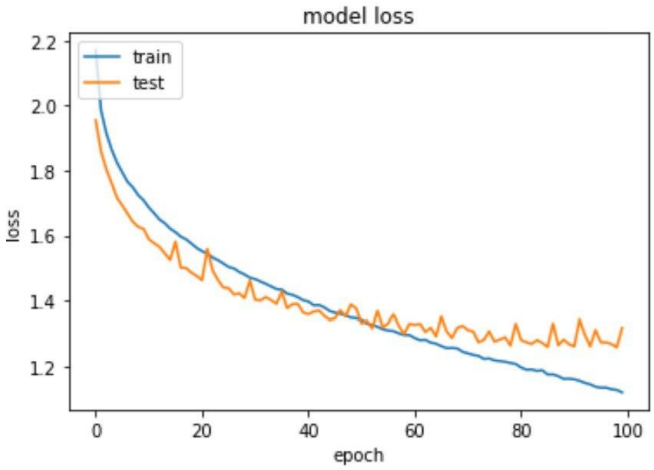
Layer (type)	Output Shape	Param #
dense_20 (Dense)	(None, 1024)	3146752
activation_20 (Activation)	(None, 1024)	0
dropout_15 (Dropout)	(None, 1024)	0
dense_21 (Dense)	(None, 512)	524800
activation_21 (Activation)	(None, 512)	0
dropout_16 (Dropout)	(None, 512)	0
dense_22 (Dense)	(None, 512)	262656
activation_22 (Activation)	(None, 512)	0
dropout_17 (Dropout)	(None, 512)	0
dense_23 (Dense)	(None, 10)	5130
activation_23 (Activation)	(None, 10)	0

شکل ۵- مشخصات مدل **MLP**

نمودار های accuracy و loss این مدل را روی داده های تست در شکل ۶ و ۷ مشاهده می‌کنید. مدل به دقت آموزش در حدود ۰.۶ می‌رسد در حالیکه دقت مدل روی داده‌های تست در حدود ۰.۵۵ می‌ماند و نوسان می‌کند.

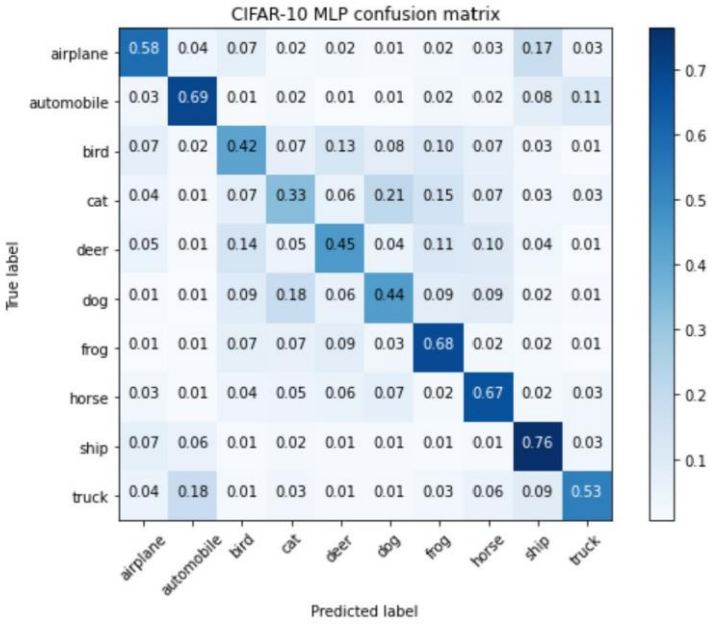


شکل ۶- نمودار **accuracy** مدل **MLP** روی دیتاست **cifar-10**



شکل ۷- نمودار **loss** مدل **MLP** روی دیتاست **cifar_10**

همچنین ماتریس آشفتگی این مدل را روی داده‌های تست در شکل ۸ مشاهده می‌کنید.



شکل ۸- ماتریس آشفتگی مدل **MLP** روی دیتاست **cifar_10**

نتیجه‌ی عملکرد مدل MLP روی دیتاست cifar_10 را در شکل ۹ مشاهده می‌کنید.

```
CL Report:
                precision    recall  f1-score   support

      airplane       0.62      0.58      0.60      1000
    automobile       0.67      0.69      0.68      1000
          bird       0.45      0.42      0.43      1000
           cat       0.40      0.33      0.36      1000
          deer       0.51      0.45      0.48      1000
           dog       0.47      0.44      0.46      1000
          frog       0.55      0.68      0.61      1000
         horse       0.59      0.67      0.62      1000
          ship       0.60      0.76      0.67      1000
         truck       0.66      0.53      0.59      1000

      accuracy                           0.56     10000
     macro avg       0.55      0.56      0.55     10000
  weighted avg       0.55      0.56      0.55     10000
```

شکل ۹- عملکرد مدل **MLP** روی دیتاست **cifar_10**

پیاده‌سازی مدل BeiT در نوت‌بوک 2_3_BeiT.ipynb آورده شده. جزئیات شبکه BeiT را با استفاده از کلاس NeuralNetwork در شکل ۱۰ می‌توان دید.

```python
pretrained_model_name = "microsoft/beit-base-path16-224-pt22k"
nn_model = AutoModelForSemanticSegmentation.from_pretrained(
    pretrained_model_name, id2label=id2label, label2id=label2id
)

class NeuralNetwork(nn.Module):
    def __init__(self):
        super(NeuralNetwork, self).__init__()
        self. nn_model = nn_model
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(5760, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10),
        )

    def forward(self, **kwargs):
        x = self. nn_model(pixel_values=kwargs['pixel_values'])
        logits = x.logits
        output = torch.sigmoid(logits)[0]
        output = output.view(2,-1)
        logits = self.linear_relu_stack(output)
        logits = torch.softmax(logits,dim=-1)
        return torch.mean(logits), _
```

شکل ۱۰- مدل **BeiT**

به علت محدودیت حافظه و زمان اجرا، شبکه را با batch_size = 2 و طی ۵ ایپاک آموزش دادیم. همین میزان کم ایپاک و سایز بچ، حدود ۳۰ دقیقه زمان برد. همچنین پارامترهای استفاده شده برای آموزش شبکه را در شکل ۱۱ مشاهده می‌کنید.

```
training_args = TrainingArguments(
    output_dir="segformer-b0-scene-parse-150"
    learning_rate=6e-5,
    num_train_epochs=5,
    per_device_train_batch_size=2,
    per_device_eval_batch_size=2,
    save_total_limit=3,
    evaluation_strategy="steps",
    save_strategy="steps",
    save_steps=20,
    eval_steps=5,
    logging_steps=1,
    eval_accumulation_steps=5,
    remove_unused_columns=False,
    push_to_hub=False,
)
```

شکل ۱۱- پارامترهای استفاده شده برای آموزش شبکه **BeiT**

۲-۴. پرسش‌ها

۱.

درCNN مفهوم کانوولوشن زدن (اعمال فیلتر) با مفهوم attention قرابت دارد چرا که زیر الگو های مهم و تکرار شونده را پیدا میکند و آنها را عبور میدهد، به عبارتی انگار توجه(attention) را به آنها جلب میکند .

۲.

مفهوم local attention به دسته ای از داده ها اشاره دارد که در نزدیکی یکدیگر قرار دارند در حالی که global attention به همجواری داده ها توجه نمی کند و attention را بر روی تمام داده ها اعمال می کند. CNN نیز معادل حالت hard-attention است که local-attention را به صورت سخت گیرانه تر با kernel کوچک تر انجام می دهد.

۳. درستی یا نادرستی

۱. نادرست

۲. درست

۳. درست

۴. نادرست