סיווג נתוני טקסט

בפרק זה נאמן רשת פשוטה ביותר לזיהוי הרגש הבא לידי ביטוי במשפט מאוסף הנתונים SST2. בפרקים הבאים נשלב בתוכה את רכיבי הארכיטקטורה המתקדמים תוך כדי פיתוחם. בדומה לבעיית סיווג הנקודות השחורות והלבנות לשתי מחלקות, המוכרת לנו מיחידה 2, גם במקרה הנוכחי אנו רוצים לבצע סיווג לשתי מחלקות. על כן, בסופה של הרשת יהיה עלינו למקם ראש סיווג המייצר פלט בעל שני איברים ומעביר אותו לפונקציית ה-Softmax, אלו יהיו הסתברויות השייכות לשתי המחלקות. בהתאם, ראש הסיווג יהיה מוגדר כלהלן.

```
class ClassificationHead(nn.Module):
    def __init__(self, in_features):
        super().__init__()
        self.linear = nn.Linear(in_features, 2)
        self.logsoftmax = nn.LogSoftmax(dim=0)

    def forward(self, feature_extractor_output):
        class_scores= self.linear(feature_extractor_output)
        logprobs    = self.logsoftmax(class_scores)
        return logprobs
```

ראו כי פלט ראש הסיווג הוא לוגריתם הסתברויות הסיווג וזכרו כי נשתמש בו עבור פונקציית המחיר האנטרופיה הצולבת.

כעת עלינו להגדיר את מחלץ המאפיינים מהמשפט, ונרצה לעשות זאת בצורה הפשוטה ביותר: נזין ישירות את פלט שכבת השיכון אל ראש הסיווג. אם כן, מחלץ המאפיינים יורכב מהשיכון בלבד ובפרקים הבאים נרחיב את הפונקציונליות שלו.

```
class FeatureExtractor(nn.Module):
    def __init__(self, embed_dim):
        super().__init__()
        self.embedding = nn.Embedding(len(vocab),embed_dim)

    def forward(self, sentence_tokens):
        embedded    = self.embedding(sentence_tokens)
        return embedded
```

בהנתן משפט לאחר טוקניזציה והמרת הטוקנים למספרים הסידוריים שלהם, מחלץ המאפיינים יחזיר טנזור בעל שני מימדים: המימד הראשון יציין את מיקום הטוקן במשפט, והמימד השני את מרחב השיכון. ראו לדוגמה,

```
print(example_sentence)

preprocess = lambda x: torch.tensor(vocab(x.split()))
tokens = preprocess(example_sentence)
print(tokens)

extractor = FeatureExtractor(2)
features  = extractor(tokens)
print(features,features.size(),sep="\n")
```

פלט:

```
contains no wit , only labored gags
tensor([2924,   61,  330,    2,   89, 1993,  549])
tensor([[ 0.9569, -0.6598],
        [ 0.9742, -1.0970],
        [-0.3451,  0.7615],
        [-0.8656,  1.8823],
        [ 0.6175,  0.2272],
        [ 0.9894, -0.8952],
        [ 1.0751, -1.2522]], grad_fn=<EmbeddingBackward0>)
torch.Size([7, 2])
```

פלט זה, המשתנה בגודלו בהתאם לאורך המשפט, עתיד ליצור התנגשות עם הקלט הצפוי לראש הסיווג: בעת יצירתו יש להגדיר את מספר נוירוני הקלט בשכבה הליניארית, זהו הפרמטר `in_features`. בכדי להתמודד עם קושי זה, נסכום את כל שיכוני הטוקנים במשפט נתון, ליצירת טנזור אשר מימדו קבוע, כמימד מרחב השיכון. השינוי הדרוש במתודת ה-`forward` של המחלץ מופיע בקטע הקוד הבא.

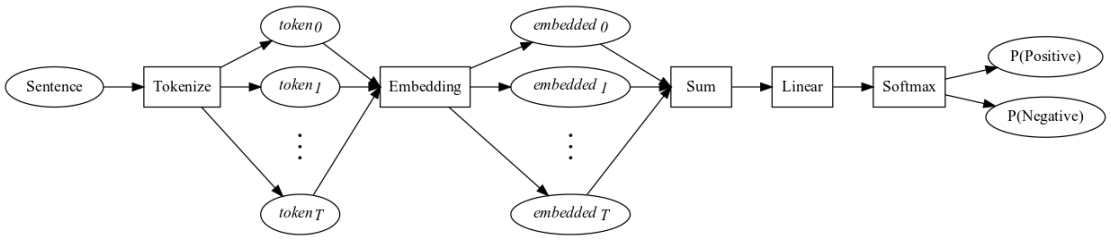
```
    def forward(self, sentence_tokens):
        embedded     = self.embedding(sentence_tokens)
        feature_extractor_output = embedded.sum(dim=0)
        return feature_extractor_output
```

כעת, כאשר מימד הפלט של מחלץ המאפיינים ידוע מראש, ניתן לחברו לראש הסיווג ולקבל את הרשת השלמה.

```
class EmbedSumClassify(nn.Module):
    def __init__(self, embed_dim):
        super().__init__()
        self.extractor  = FeatureExtractor(embed_dim)
        self.classifier = ClassificationHead(embed_dim)

    def forward(self, sentence_tokens):
        extracted_features = self.extractor(sentence_tokens)
        logprobs     = self.classifier(extracted_features)
        return logprobs
```

תהליך עיבוד הנתונים והזנתם לרשת לצורך קבלת הסתברויות הסיווג מאויר להלן.



רשת זו אינו ערוכה לקבל batch של משפטים באורכים שונים, אך תוך הזנת המשפטים אחד לאחר השני, נוכל לאמנה כרגיל.

בעוד שרשת בסיסית כנ"ל מסוגלת ללמוד לסווג את המשפטים באופן סביר, היא סובלת ממגבלה קריטית: פעולת הסכום מבטלת את משמעות סדר הופעת המילים במשפט. ראו בדוגמה הבאה כיצד שני משפטים הנבדלים רק בסדר הופעת המילים מסווגים באופן זהה.

```
example_sentences=["very good , not bad",
                   "very bad , not good"]
with torch.no_grad():
  for sent in example_sentences:
    print(preprocess(sent))
    print(torch.exp(model(preprocess(sent))))
```

פלט:

```
tensor([77, 46,  2, 33, 74])
tensor([1.0000e+00, 4.7996e-07])
tensor([77, 74,  2, 33, 46])
tensor([1.0000e+00, 4.7996e-07])
```

דבר זה אינו מפתיע, שכן הטוקנים של שני המשפטים זהים, ועל כן הקלט לראש הסיווג עבור שניהם זהה, וישאר כך גם לאחר האימון.

שאלות לתרגול

1. הגדירו אובייקט `EmbedSumClassify`, הזינו את משפטי סט האימון לרשת והעבירו את הפלט המתקבל לפונקציית המחיר האנטרופיה הצולבת. אמנו את הרשת לסיווג הרגש המובע במשפטים תוך הזנת המשפטים אחד-אחד. בדקו את ביצועי הרשת המאומנת עבור מימדי שיכון שונים.
2. החליפו את שכבת השיכון ופעולת הסכום בשכבת `nn.EmbeddingBag`, המבצעת שתי פעולות אלו אחת אחרי השניה באופן יעיל הרבה יותר ואמנו את המודל מחדש. השוו את זמני הריצה. שימו לב שהקלט הצפוי של שכבה זו הוא batch של משפטים, ולכן בעת הזנת משפט יחיד עליכם להוסיף מימד מנוון בעזרת המתודה `unsqueeze`, ולהוריד אותו בהמשך.
3. החליפו את שכבת השיכון בשיכוני GloVe והשוו את התוצאה לרשת עם שיכונים נלמדים. השוו גם את זמני הריצה של תהליך האימון ונסו להסביר את הפער.