Farshad Borjalizade

HW6 – Arooz Classification

Natural Language Processing

University of Tehran

Shahrivar 1400

گزارش کار انجام شده

هدف اصلی در این تمرین پیاده سازی یک طبقه بند کننده (classifier) برای تشخیص وزن های ابیات اشعار است. مجموعه داده ای که در اختیار داریم شامل فایل ابیات فارسی(DSVN)، تبدیل شده ابیات به کاراکترهای cv (DSCV) و یک فایل که تبدیل شده به کاراکترهای U- (DSUS) و فایل وزن های هر بیت (DSA) است، که برای هر کدام از آنها فایل های آموزش از تست جدا شده است.

برای پیش پردازش داده ها به این صورت عمل کردیم که با استفاده از تابع pre_procces() ابتدا فایل های مربوط به کاراکترهای cv و U- را به همراه برچسب های آنها را می خوانیم و داده هایی را که در فایل به صورت درستی قرار ندارند یا اطلاعات ناقصی را در بر دارند را حذف می کنیم، که نتیجه در تصویر زیر به صورت جدولی نمایش داده شده است.

	cv1	cv2	m1	m2	label
0	cvccvvccvcvccvcccvcvvcvvccvcvvcvcvc	cvvcvcccvcvccvvcvccvvcvccvcvvccvvc	--UU--UU--UU--	--UU--UU--UU--	1003
1	cvvccvvcvccvcvcvvcvcccvcvccvvccvcvc	cvvvcvvcvvcvccvccvvcvcvvcvccvcvc	UU--UU--UU--UU-	UU--UU--UU--UU-	1001
2	cvccvcvccvcccvcvccvccvcvccvccvcvcvvc	cvccvcvvccvcvvcvccvcvcvvcvvccvcvc	UU--UU--UU--UU-	UU--UU--UU--UU-	1001
3	cvcvcccvccvccvcvccvcccvccvcvccvccvvc	cvvcvvccvvcvccvvcvccvvccvvcvcvvcvcvvcvc	-U---U---U---U-	-U---U---U---U-	1014
4	cvcvcvvcvcvccvcvccvccvcvcvvcvvccvcvccvc	cvcvcvccvccvccvcvccvvvcvcvvcvcvcvcvvc	UU--UU--UU--UU-	UU--UU--UU--UU-	1025
...	...	...	...	...	...
919	cvccvcvvcvccvcvcvvcvvccvcvcvcvvccvcvv	cvvcvcvccvcvccvcvcvccvcvccvcvcvcvvc	UU--UU--UU--UU-	UU--UU--UU--UU-	1001
920	cvccvcvcvcvvccvcvccvcccvccvccvvccvcvc	cvvcvccvvcvccvcvcvccvvcvccvvcvccvcvc	-UU--UU--UU--UU-	-UU--UU--UU--UU-	1024
921	cvccvcvcvcccvcvccvvcvcvcvccvccvcvc	cvcvcvccvcvcvcvcvvcvvccvcvcvccvccvvcvc	-UU--U--UU--U-	-UU--U--UU--U-	1001
922	cvvcvvcvcvccvcvvvcvvcvccvvcvccvcvccvc	cvccvvcvcvccvcvvcvcvcccvcvccvcvccvc	--U-U----U-U--	--U-U----U-U--	1007
923	cvcvcvccvcvvccvvvcvcvvcvvcvccvvccvv	cvvcvvcvvcvcvvcvvvcvvcvvccvvcvcvvc	--U-U-UU--U-U-	--U-U-UU--U-U-	1003

910 rows × 5 columns

همانظور که از اندیس سطرها مشخص است تعدادی از سطرهایی که داده های مناسبی نداشتند حذف شده است.

از آنجایی که داده های ما به صورت کاراکتری هستند می بایستی آنها را به صورت برداری به شبکه دهیم، برای تبدیل کاراکترها به بردار این رویکرد را در نظر گرفتیم که هر کدام از کاراکترها را به یک عدد منحصر به فرد نگاشت کنیم به این صورت که کاراکتر – را به ۴، کاراکتر U را به ۳، کاراکتر c را به ۲ و کاراکتر v را به ۱ نگاشت می کنیم.

بر اساس تجربه و کم بودن تعداد داده با این رویکرد داده ها را آماده آموزش کردیم که ستون cv1 را با ستون m1 و ستون cv2 را با ستون m2 در نظر گرفتیم با این کار تعداد بردارهای آموزش مان دو برابر می شود. برچسب ها را هم به صورت one-hot به یک بردار با طول ۳۱ که تعداد کلاس هایمان تبدیل می کنیم. که تمام این تبدیلات در تابع mapping() انجام می گیرد.

برای آموزش داده ها از یک شبکه fully connected استفاده کردیم که خلاصه ای از آن را مشاهده می کنیم.

Layer (type)	Output Shape	Param #
input_22 (InputLayer)	[(None, 65)]	0
dense_210 (Dense)	(None, 1024)	67584
dense_211 (Dense)	(None, 1024)	1049600
dense_212 (Dense)	(None, 512)	524800
dense_213 (Dense)	(None, 512)	262656
dense_214 (Dense)	(None, 512)	262656
dense_215 (Dense)	(None, 256)	131328
dense_216 (Dense)	(None, 256)	65792
dense_217 (Dense)	(None, 128)	32896
dense_218 (Dense)	(None, 128)	16512
dropout_21 (Dropout)	(None, 128)	0
dense_219 (Dense)	(None, 31)	3999

Total params: 2,417,823
Trainable params: 2,417,823
Non-trainable params: 0

از بین بهینه سازها بهترین نتیجه را با استفاده از Stochastic gradient descent (SGD) **با پارامترهای** learning_rate=0.001, momentum=0.98, decay=0.0001 **گرفتیم که بعد از** ۲۰۰ epoch **به دقت** ٪۵۴ **رسیدیم.**