دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)

گزارش‌کار پروژه درس یادگیری ماشین کاربردی

نگارش

مریم سلطانی

(دانشجوی رشته مهندسی مکانیک)

استاد درس

دکتر احسان ناظرفرد

مرداد۱۴۰۱

فهرست مطالب

فهرست اشکال

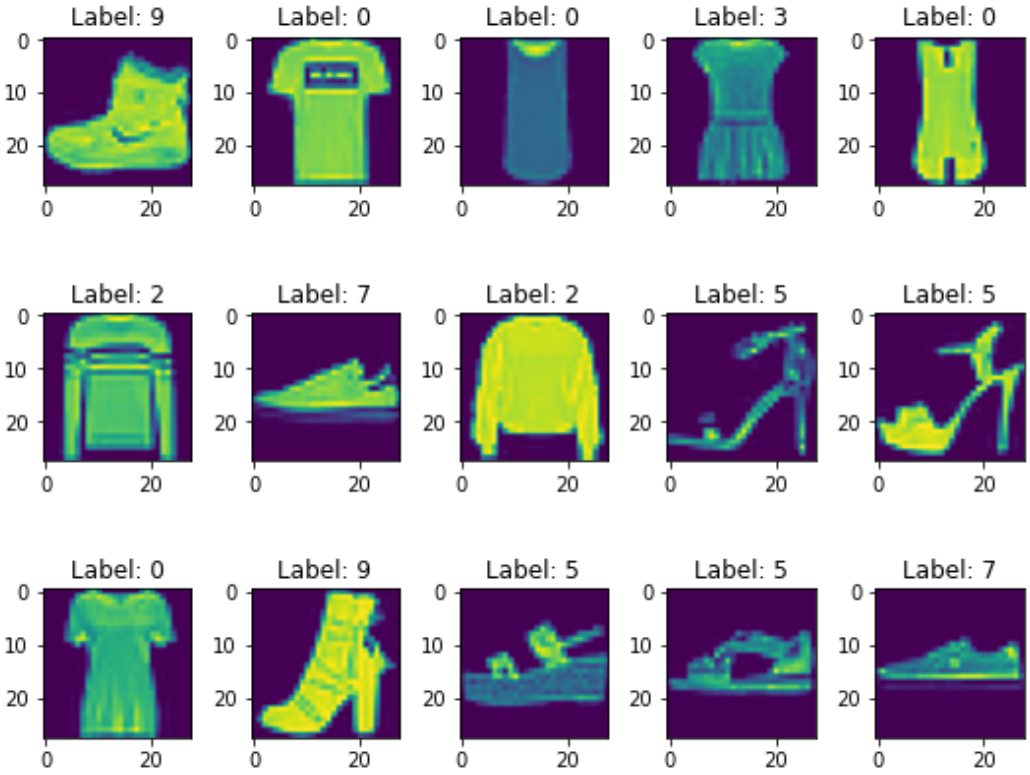
فهرست جداول

بخش اول

مجموعه داده MNIST Fashion از ۶۰ هزار تصویر آموزش و ۱۰ هزار تصویر ارزیابی تشکیل شده است. هر تصویر در این مجموعه داده دارای ابعاد ۲۸ ×۲۸ (پیکسل) است و متعلق به یکی از ۱۰ کلاس مختلف پوشاک می باشد. عنوان کلاس‌های این مجموعه داده در جدول ۱ قابل مشاهده است.

جدول۱- عنوان کلاس‌ها در مجموعه داده MNIST Fashion

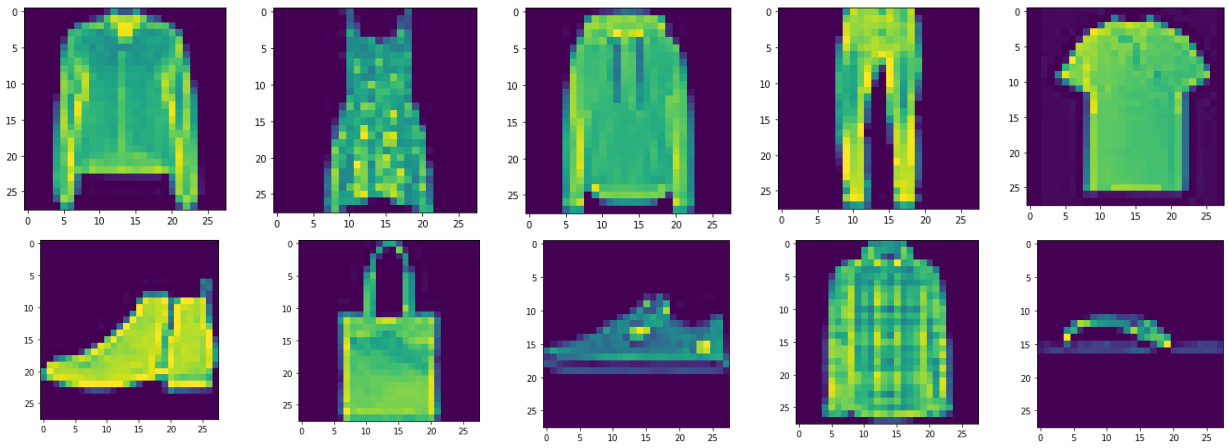
Label	Class
0	T_shirt /Top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle Boot



شکل۱- نمایش اولیه ۱۵ عکس در مجموعه داده mnist fashion

برای تعیین نمای اولیه از نوع داده‌ها در دیتاست، شکل۱ که نشان دهنده ۱۵ عکس از این مجموعه به همراه لیبل آنهاست ترسیم شده است.

پس از ایمپورت کردن داده‌ها در گوگل کلب[1] با دستور tf. Keras. Datasets. fashion_mnist در گام ابتدایی به تصادف از هر کلاس یک تصویر انتخاب شده است. برای انجام این کار، ابتدا تعداد عکس‌ها در هر کلاس به کمک لیبل آن‌ها شمارش شده است. نتیجه آنکه به ازای هر یک از ۱۰ کلاس موجود در مجموعه داده، ۶۰۰۰ عکس وجود دارد که مجموع آن‌ها ۶۰۰۰۰ عکس خواهد بود. سپس با کمک np.random.randint در ۱۰ مرحله، هر بار یک عدد تصادفی در محدوده ۰تا تعداد عکس موجود در هر کلاس تولید شده است. حال این ده عدد تصادفی باید به مجموعه داده نسبت داده شوند تا عکس‌های مورد نظر پیدا شود. بدین منظور، از اولین عکس دیتاست شمارش آغاز شده و شماره عکس(بین ۶۰۰۰۰عکس) که نشان دهنده عدد تصادفی مشخص شده در کلاس معین بوده است در متغیر هدف قرار گرفته است تا قابل فراخوانی باشد. حال این عکس‌های تصادفی قابل تعیین هستند که به ترتیب در شکل ۲ قابل مشاهده می‌باشند.



شکل۲- انتخاب تصادفی در میان ده کلاس مختلف

قبل از تنظیم لایه‌های شبکه عصبی، به کمک one hot encoder لیبل‌ها به فرمت یکسانی درآمده‌اند. این عمل در پیش پردازش میتواند تاثیر خوبی در خروجی نهایی داشته باشد. پس از گذر از این مرحله مدلی کاملا متصل[2] با دستور sequential تولید شده است. در لایه ابتدایی هر عکس با دستور flatten از فرمت ۲۸ ×۲۸ خارج شده و برای لایه ابتدایی تعداد ۲۸ ×۲۸ عدد نورون درنظر گرفته شده است. سپس برای لایه‌های میانی، دو لایه

[1] Google Colaboratory
[2] Fully Connected

عمیق[3] لحاظ شده است که به ترتیب ۵ و۳ نورون دارند. تعداد لایه‌های عمیق و تعداد نورون موجود در آن‌ها همگی هایپرپارامتر میباشد. نوع تابع فعالساز نیز relu انتخاب شده است. چون الگوهای موجود خطی نیستند برای خارج کردن نتیجه از حالت خطی از این تابع استفاده شده است. گزینه‌های دیگری نیز برای تابع فعالسازی مثل tanh و سایر موارد وجود دارند اما بنا به نتایج خوب و قابل توجهی که در سال‌های اخیر، متوجه این تابع بوده است برای این کار انتخاب شده است. لایه خروجی شامل ۱۰ نورون و با تابع فعالساز softmax میباشد. عملا در خروجی دسته‌بندی بین این ده حالت مختلف صورت میگیرد. از بین انواع optimizer های موجود Adam انتخاب شده است و برای loss function، انتخاب با توجه اینکه نوع مساله دسته‌بندی است categorical_crossentropy بوده است. خلاصه آنچه برای مدل اولیه در نظر گرفته شده است در شکل۳ موجود است.

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_5 (Flatten)         (None, 784)               0

 dense_19 (Dense)            (None, 5)                 3925

 dense_20 (Dense)            (None, 3)                 18

 dense_21 (Dense)            (None, 10)                40

=================================================================
Total params: 3,983
Trainable params: 3,983
Non-trainable params: 0
_________________________________________________________________
```

شکل۳- خلاصه مشخصات مدل انتخاب شده اولیه

حال پس از اطمینان از عملکرد مناسب شبکه، بنا بر خواسته مساله در حداقل شش آزمایش جداگانه تعداد نورون و لایه میانی تغییر پیدا کرده است تا با مقایسه آن‌ها بتوان بهترین نتیجه موجود در بین این تغییرات را پیدا کرد. تعداد گام تکرار(Epoch) ۱۰۰ لحاظ شده است. نتایج در جدول ۲ قابل مشاهده است. برای انتخاب معیار اولیه accuracy حداقل ۰/۷ مدنظر قرار داشته است.

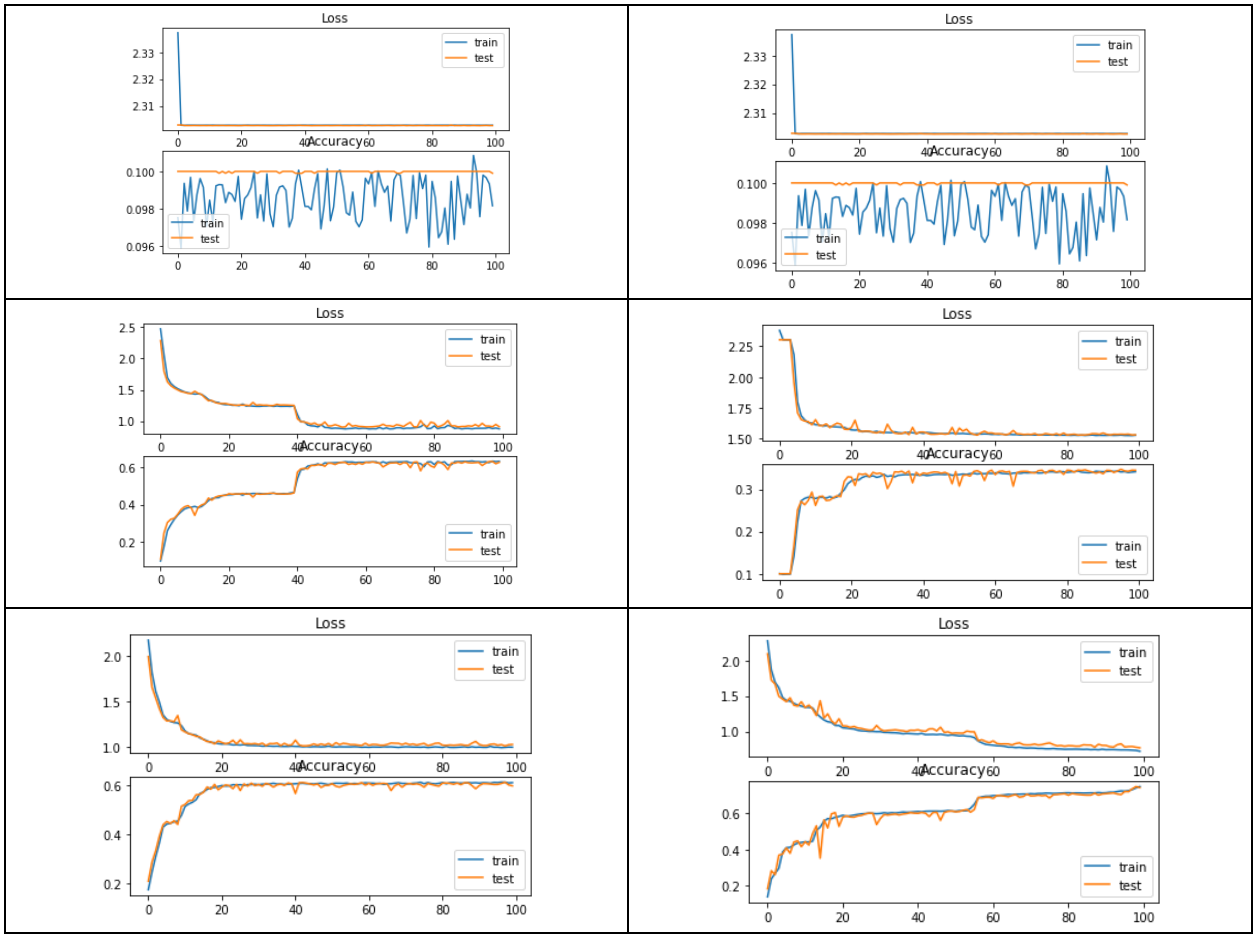
لازم به ذکر است تجربه نشان داد با ران کردن مجدد هر مدل نتایج تا حد کمی تفاوت میکند که به تعیین پارامترها و همگرایی مربوط است. مقایسه دقیق‌تر آزمایشات انتخاب شده در گام اول میتواند نتیجه نهایی را مشخص کند. مشخصات مدل انتخابی نهایی بین مدل‌های آزمایش‌شده در جدول ۳مشخص شده است.

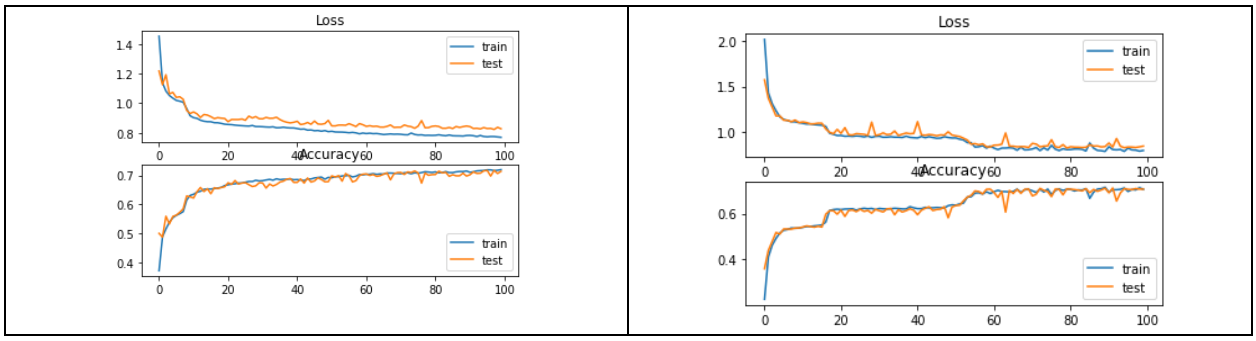
[3] Dense

جدول۲- بخشی از آزمایشات انجام شده برای دسته‌بندی دیتاست fashion mnist

شماره آزمایش	تعداد نورون در هر لایه میانی(به ترتیب از چپ به راست)	پذیرش بر اساس معیار اولیه
۱	(۵ - ۳)	خیر
۲	(۳ - ۳)	خیر
۳	(۵ - ۵)	خیر
۴	(۵ - ۵-۵)	بله
۵	(۵ - ۷-۵)	بله
۶	(۵ - ۷-۷)	خیر
۷	(۵ -۷- ۵-۵)	خیر
۸	(۵ -۵- ۵-۵)	خیر
۹	(۵ -۶- ۷-۸)	بله

نتایج در قالب نمودار accuracy-loss برای تمامی آزمایشات قابل مشاهده است.

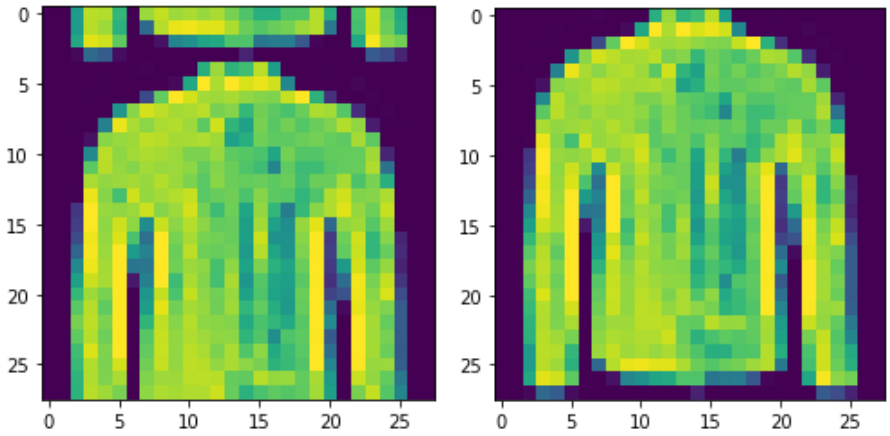




شکل ۴- نمودار accuracy-loss در آزمایشات مختلف

جدول۳- مشخصات مدل انتخابی

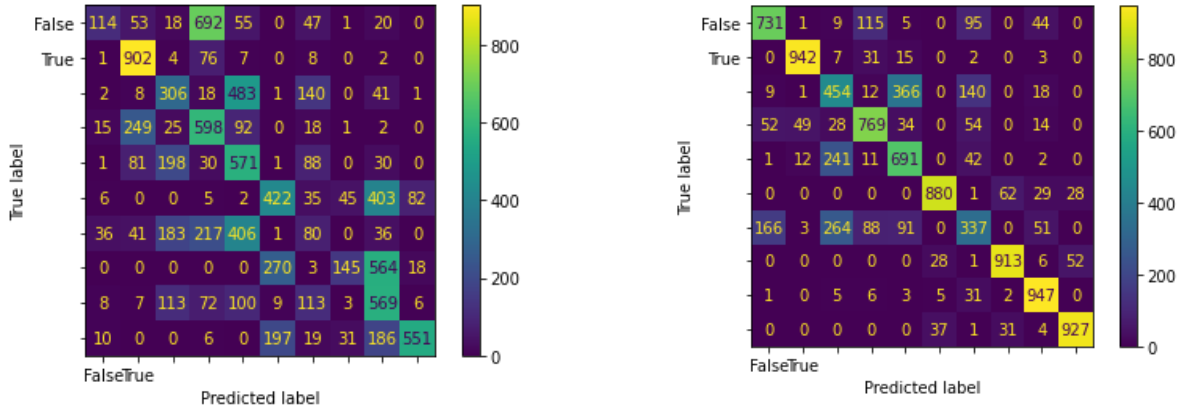
شماره آزمایش	Val_acc	Val_loss	loss	accuracy
۹	۰/۷۵۲۷	۰/۶۵۶۹	۰/۶۱۸	۰/۷۶۱۳



شکل ۵- نمونه‌ای از تصاویر بعد از اعمال شیفت ۴ واحدی به بالا

سمت راست: قبل از شیفت و سمت چپ: بعد از شیفت

برای شیفت ۴ واحدی تصاویر از numpy.roll استفاده شده است که روی تمام عکس‌های بخش تست اعمال شده است. مشاهده میشود که پس از شیفت تصاویر، تمامی تصاویر مشابه شکل ۵ تغییر خواهند یافت. خروجی یک شبکه عصبی، قبل و بعد این شیفت مقایسه شده است و نشان از کاهش چشمگیر دقت دارد که در شکل ۶ قابل مشاهده است. این شکل ماتریس در هم ریختگی برای ده کلاس را نشان میدهد که بعد از انجام شیفت تعداد حدس‌های درست شبکه به شدت کاهش یافته است.



شکل۶- ماتریس در هم ریختگی شبکه عصبی

سمت راست قبل از شیفت و سمت چپ بعد از شیفت

لازم به ذکر است انتخاب آزمایش‌ها از هیچ الگوی مشخصی پیروی نمیکند. با هرطور تغییر آزمایش‌ها مثل تغییر تعداد لایه میانی یا نورون میتوان نتایج متفاوتی گرفت و بینهایت آزمایش محتمل است. به عنوان مثال نتایج یک آزمایش دیگر که خروجی بهتری نسبت به موارد قبل داشته در شکل۶ و جدول ۳ قابل مشاهده است. این آزمایش ۴لایه میانی دارد که به ترتیب۸، ۷، ۶، و۵ نورون دارند. آزمایشات دیگری هم انجام شده که از ذکر نتایج آن برای جلوگیری از پراکندگی مطالب اجتناب شده است. ضمنا نمودار مقدار accuracy و loss برای مدل انتخابی در بخش پیوست موجود است.

قابلیت‌هایی در شبکه‌های عصبی کانولوشنی موجود است که پیشنهاد میشود برای کارهایی که با عکس سر و کار دارند از این نوع شبکه استفاده شود. یکی از این مواردtranslation invariance میباشد که به موجب آن شبکه علاوه بر تصاویری که برای آن ران شده است برای تصاویری که شیفت پیدا کرده‌اند هم کار میکند. عملا شیفت مکانی نادیده گرفته میشود. این مساله بواسطه حضور لایه‌های کانولوشنی و لایه‌های ماکس پولینگ رخ میدهد. مثلا لایه‌های کانولوشنی تصویر را به مجموعه‌ای از فیچرها و موقعیت‌های نسبی آنها کاهش میدهد سپس ماکس پولینگ رزولوشن و پیچیدگی این مورد را کاهش میدهد. تصاویری در قسمت پیوست برای این مورد موجود است. البته میزان بالای شیفت ممکن است نیاز به استفاده از لایه‌های کانولوشنی و ماکس پولینگ بیشتری داشته باشد.

از این جهت شبکه حاضر بعد از اعمال شیفت خیلی موفق عمل نکرده است که دور از ذهن هم نیست.

بخش دوم

ابتدا مجموعه داده weatherAUS.csv در محیط گوگل کلب خوانده شده است. همه ستون‌ها بجز ستون آخر در ماتریس x ذخیره شده‌اند. ستون آخر همان مقدار y خطاب خواهد شد که عملا خروجی را مشخص میکند. این مجموعه داده شامل ۱۴۵۴۶۰ سطر است و به طور کلی ۲۳ ستون دارد. در گام بعدی بررسی شده که فیچرهای غیر عددی[۴] و عددی[۵] کدام هستند. لیست این خصوصیات در جدول ۴ قابل مشاهده میباشد.

جدول۴- مشخصه‌های عددی و غیرعددی در مجموعه دادهweatherAUS

خصوصیات غیر عددی	'Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'RainToday', 'RainTomorrow'
خصوصیات عددی	'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine' و 'WindGustSpeed','WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm'

حال لازم است بررسی شود که آیا داده تعریف نشده در این دیتاست وجود دارد یا خیر. خلاصه این بررسی نشان میدهد که تعداد داده‌های تعریف نشده به ازای هر یک از خصوصیات مطابق جدول۵ میباشد.

جدول۵- تعداد داده‌های تعریف‌نشده به ازای خصوصیات مختلف

نام خصوصیت	تعداد nan	types	نام خصوصیت	تعداد nan	types
Location	0	object	WindSpeed3pm	3062	Float64
MinTemp	1485	Float64	Humidity9am	2654	Float64
MaxTemp	1261	Float64	Humidity3pm	4507	Float64
RainFall	3261	Float64	Pressure9am	15065	Float64
Evaporation	62790	Float64	Pressure3pm	15028	Float64
Sunshine	69835	Float64	Cloud9am	55888	Float64
WindGustDir	10326	Object	Cloud3pm	59358	Float64
WindGustSpeed	10263	Float64	Temp9am	1767	Float64
WindDir9am	10566	object	Temp3pm	3609	Float64
WindDir3pm	4228	Object	RainToday	3261	Object
WindSpeed9am	1767	Float64	RainTomorrow	3267	Object

[4] Categorical
[5] Numerical

با توجه به مشخص بودن تعداد کل داده‌ها امکان محاسبه درصد داده تعریف نشده به ازای هر فیچر هم موجود است. آخرین خصوصیت که با رنگ متفاوت نشان داده شده است همان خروجی هدف میباشد. سطرهایی که خروجی نهایی در آنها مشخص نیست حذف شده‌اند زیرا مقدار هدف در آنها معلوم نیست و عملا داده‌های این سطرها نمیتواند مفید باشد. برای ترمیم آن دسته از فیچرهایی که غیر عددی هستند، داده‌های تعریف‌نشده با پرتکرارترین داده جایگزین خواهند شد. تنوع متغیرها در متغیرهای غیر عددی برحسب تعداد در جدول ۶ قابل مشاهده است.

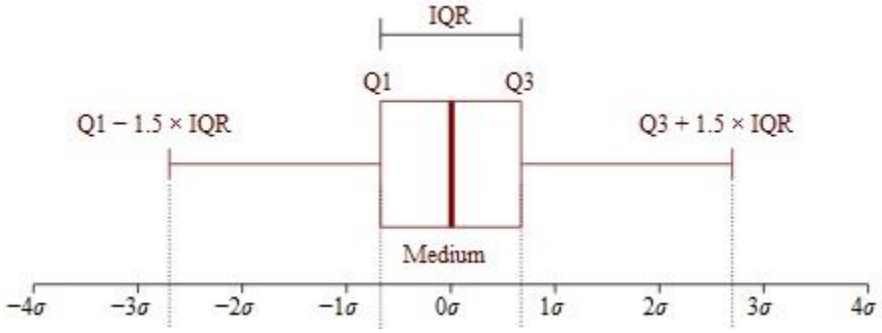
جدول۶- تنوع کلاس‌های متغیرهای غیرعددی

Date	RainTomorrow	RainToday	WinDir3pm	WinDir9am	WindGusdir	Location
۳۴۳۶	۲	۲	۱۶	۱۶	۱۶	۴۹

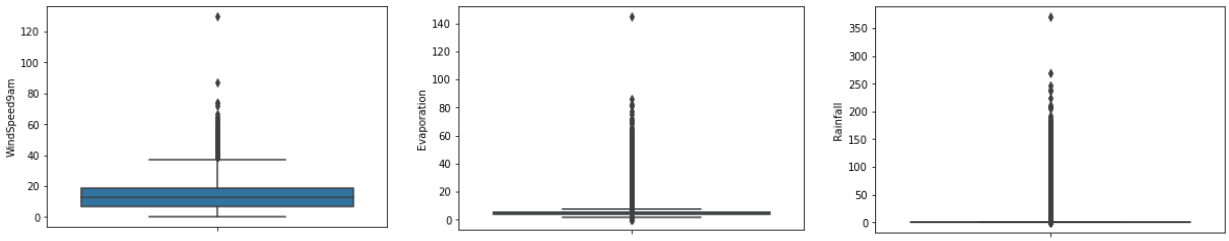
می‌توان در جایگذاری متغیرهای غیر عددی از سری اعداد متوالی استفاده کرد مثلا برای حالتی که ۴ مقدار مختلف برای یک فیچر وجود دارد از اعداد ۱تا۴ استفاده کرد و به هر کلاس یک عدد نسبت داد. اما این عمل درستی نیست و باید از مسیر onehot جلو رفت. یعنی در لحظه فقط مقدار متغیر روشن باشد و سایر حالات خاموش باشد. عملا پایتون بین بزرگی و کوچکی اعداد تفاوت قائل میشود. این کار باعث می‌شود تعداد ستون‌های فیچرها یکباره به طور قابل توجهی افزایش یابد چرا که مثلا اگر سه مقدار برای متغیری در دیتاست وجود داشته حال نیاز به سه ستون مجزا برای آن است که عملا سه فیچر جلو میکند. برای تکمیل دیتاست در قسمت داده‌های عددی از median استفاده شده که حساسیت کمتری به داده پرت دارد و در جایگذاری مقادیر تعریف نشده از متغیرهای غیر عددی از mode کمک گرفته شده است. مطابق شکل۷ با مقایسه مشاهده میشود که تعداد داده‌های خروجی دو گروه yes وno با هم متناسب نیستند و گروه no بسیار پرتعدادتر میباشد. اما باتوجه به آنکه این توزیع برگرفته از یک دیتاست واقعی است کار خاصی نمیتوان کرد و درست نیست که تغییرات مصنوعی ایجاد شود.

شکل۷- مقایسه دو کلاس متفاوت خروجی در دیتاست weatherAUS

در گام بعدی لازم است داده پرت تشخیص داده شود. بدین منظور باید به سراغ مفاهیم آماری رفت. روشی که در این کار استفاده شده است بدین صورت می‌باشد که دو متغیر q1 وq3 تعریف شده‌اند که به ترتیب میانه در نیمه اول و دوم دیتاست را مشخص میکنند. مقدار IQR تفاوت این دو عدد است و چنانچه مطابق شکل ۸ عددی خارج از محدوده مشخص شده باشد، به‌عنوان داده پرت شناخته میشود. لازم به ذکر است که پیش از اعمال این روش باید مطمئن شد در کدام فیچرها احتمال داده پرت وجود دارد. بدین منظور با رسم نمودار جعبه‌ای نتیجه حاصل شد که برای سه فیچر داده پرت وجود دارد. نمودار جعبه‌ای این سه فیچر در شکل۹ قابل مشاهده است.

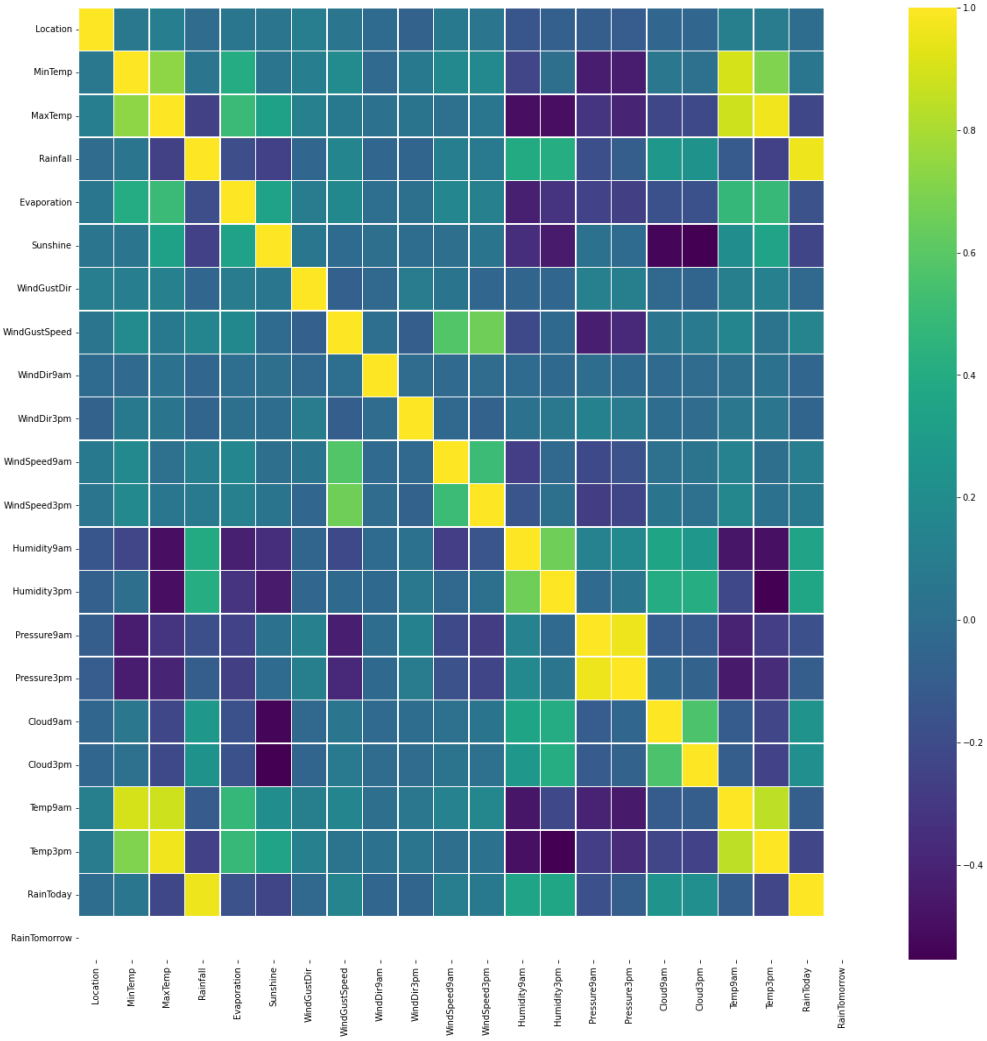


شکل۸- شماتیک معیار انتخابی برای تعیین داده پرت



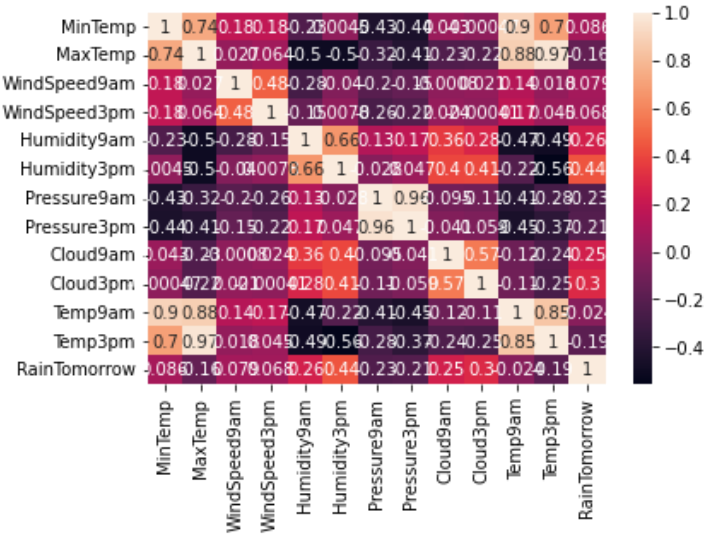
شکل ۹- نمودار جعب‌های در سه فیچر که نشان از وجود داده پرت دارد

حال باید کرلیشن فیچرهای مختلف با همدیگر را بررسی کرد چرا که تعداد متغیرهای ورودی زیاد است و شاید بتوان از این طریق آن‌ها را کم کرد. برای این کار کرلیشن متغیرها با همدیگر ترسیم شده است که در شکل ۱۰ قابل مشاهده است.



شکل۱۰- همبستگی مشخصه‌های مختلف نسبت به همدیگر

روشن‌ترین رنگ‌ها نشان‌دهنده همبستگی زیاد متغیرها میباشد. در این میان خانه‌هایی که رنگ روشن دارند به طور دقیق‌تر بررسی شده‌اند تا چنانچه ضریب همبستگی دو فیچر از عدد ۰/۹ بیشتر باشد، یکی از آن دو حذف شوند. به منظور بررسی دقیق‌تر مقایسه میتواند بر اساس شکلی مشابه شکل ۱۱صورت گرفته که اعداد به طور دقیق در آن واضح هستند یا حتی با توجه به رنگ، در صورت مشکوک بودن به مقادیر در خانه‌های شکل ۱۰ آنرا به طور دقیق فراخوانی کرد.



شکل۱۱- نمایی از مقدار همبستگی فیچرهای مختلف

با همین منطق دو فیچر Temp3pmیا max temp وPressure3pm یا pressure 9amباید حذف شوند. چرا که هرکدام به فیچر دیگر، وابستگی بالای ۰/۹ دارند.

در تقسیم داده‌ها به دو بخش آموزش و آزمون، به ترتیب ۲۰ درصد داده‌ها برای تست و ۸۰ درصد برای آموزش لحاظ شده است. حال با کمک Standardscaler اردر داده‌ها یکسان شده است. این مرحله در پیش پردازش با توجه به تفاوت رنج اعداد در فیچرهای مختلف بسیار ضروری بنظر میرسد. حال یک شبکه عصبی ساخته شده است که در گام اولیه، لایه ورودی ۱۶ نورون، در لایه میانی ۳ نورون و در لایه خروجی ۱ نورون دارد که نشان میدهد مساله رگرسیون است. اپتیمایزر[6] انتخاب شده ADAM با نرخ یادگیری ۰/۰۱ میباشد. نتایج در جدول۷ قابل مشاهده میباشد.

لازم به ذکر است که در مرحله آخر به کمک selectkbest تعداد فیچرها کاهش پیدا کرده است و مجددا نتایج طلب شده که این مورد نیز در جدول ۸ قابل مشاهده میباشد. انتخاب تعداد فیچرها به کمک این تابع اختیاری است. مثلا میتوان ۵ مورد یا بیشتر را نگه داشت. میتوان تعبیر کرد که این انتخاب، نوعی هایپر پارامتر است. یعنی ممکن است بر اساس تعداد انتخابی، نتایج مختلفی حاصل شود.

[6] Optimizer

جدول۷: خلاصه نتایج خروجی مدل برای پیش‌بینی بارش یا عدم بارش باران

classification_report:

	precision	recall	f1-score	support
0	0.85	0.97	0.91	16421
1	0.77	0.40	0.52	4525
accuracy			0.84	20946

بخش سوم

ابتدا مطابق پیشنهاد صورت پروژه کتابخانهhazm در محیط گوگل کلبinstall شده است تا بتوان از امکانات آن استفاده کرد. سپس مجموعه داده صورت سوال که یک فایل اکسل است با کمک دستور قرار داده شده در تعریف پروژه دانلود شده است و در یک دیتافریم جای گرفته است. این مجموعه داده شامل ۱۲۰۰۰سطر و دو ستون است همچنین خوشبختانه هیچ دادهnull(یا تعریف‌نشده) ندارد. ستون دوم شامل ۵ حالت مختلف عدد بین منفی ۲ تا مثبت دو میباشد که برای دید مناسب‌تر، توزیع این پاسخ‌ها شمارش شده است که مطابق جدول ۸ میباشد.

جدول۸- توزیع تعداد جملات در هر یک از کلاس‌های پنج‌گانه

کلاس	+۲	+۱	۰	-۱	-۲
تعداد جملات	۲۲۳۵	۳۸۳۲	۴۵۶۱	۱۲۱۱	۱۶۱

مدل شماره یک برای دسته‌بندی بدون انجام پیش‌پردازش ساخته شده است. ۹۴۶۹ مورد از ۱۲۰۰۰ سطر(حدود ۸۰ درصد) برای آموزش و مابقی برای تست درنظر گرفته شده است. برای این کار بر خلاف روش‌های بکار گرفته شده در سایر قسمت‌ها، یک عدد تصادفی به هر سطر نسبت داده شده است. با انتخاب یک ترشولد[۷] برای عدد تصادفی تولید شده، هر سطر در گروه تست[۸] یا ترین[۹] جای گرفته است. برای تبدیل داده‌های متنی به بردار از count vectorizer موجود در کتابخانه سایکیت لرن[۱۰] کمک گرفته شده است. اساس کار این تابع بر شمارش تمام کلمات و روشن یا خاموش‌کردن هر یک از کلمات شمارش شده برای هر جمله است. در ادامه مدل ساخته شده است که مدل ANN شامل تابع فعالساز relu و تعداد ۵ نورون در خروجی است که عملا داده‌ها را در ۵ گروه دسته‌بندی می‌کند. اپتیمایزر adam برای کار انتخاب شده است. البته انتخاب نوع اپتیمایزر نوعی هایپرپارامتر است که میتواند بهینه شود. مثلا انتخاب اپتیمایزرهای مختلف میتواند نتایج را تا حدی تغییر دهد اما در مساله حاضر به این مورد بسنده شده است. برای لایه میانی۸ و ۱۶ نورون انتخاب شده است و برای سنجش خطا هم مشابه بخش اول پروژه از crossentropy کمک گرفته شده است. نهایتا پس از ۱۰ گام، accuracy حدود ۹۱ درصد حاصل شده است که تصویر خروجی شبکه در بخش پیوست موجود است.

در مرحله دوم مدلی ساخته شده است که قبل از آن پیش‌پردازش روی داده‌ها صورت گرفته است. پیش پردازش شامل مراحل متفاوتی میتواند باشد که به چند مورد آن در صورت پروژه اشاره شده است اما سایر موارد به افراد واگذار شده است. تا آنجا که فرصت جستجو و بررسی فراهم بوده تلاش شده تا پیش پردازش با جستجو در منابع

[7] Treshhold
[8] Test
[9] Train
[10] Scikit-learn

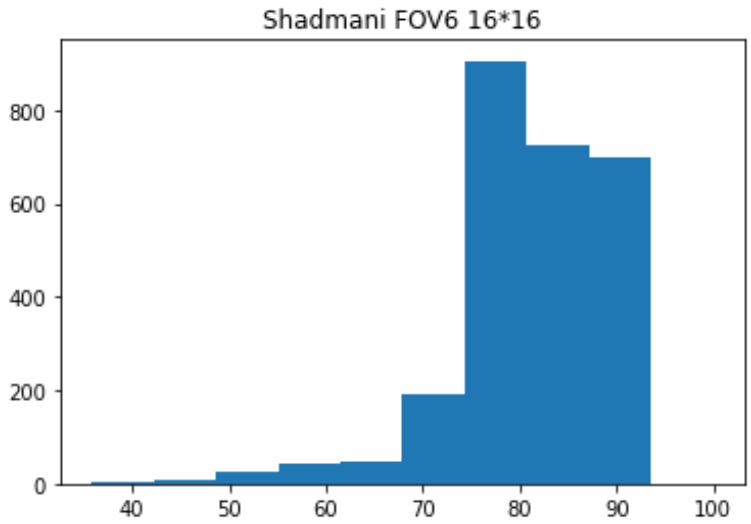
مختلف به خوبی انجام بگیرد. در گام اول اطلاعات زائدی مثل ایموجی‌ها، تگ‌های html، آدرس وبسایت و اعداد حذف شده‌اند. در مرحله بعدی علائم نگارشی[11] مانند نقطه، ویرگول، علامت تعجب و ... حذف شده است. سپس تمام حروف انگلیسی از این دیتاست فارسی حذف شده است. در گام بعدی تعدادی از کلماتی که زائد هستند و بار عملکردی مثبتی ندارند حذف شده‌اند مثل "سلام"، "بنابر" و.... این کلمات همان کلمات ایست[12] نام دارند. انتخاب این کلمات بر اساس جستجو بوده است و در این زمینه پیشنهادات مختلفی برای این کار در سایت‌های مختلف وجود دارد. گام بعدی برای جداسازی کلمات[13] میباشد. یعنی عملا هر جمله به تمام کلمات موجود در آن تقسیم میشود و کلمات هر عبارت ماهیت مستقل پیدا میکنند. دو مرحله بعدی ریشه‌یابی[14] و بن‌واژه‌سازی[15] است که در صورت پروژه نیز به آنها اشاره شده است. در مرحله آخر باید بردارسازی انجام شود که مشابه کاری است که در ساخت مدل اول انجام شد. با انتخاب سه گروه نورون ۳۲تاییبرای لایه میانی و ۵ نورون در خروجی، accuracy پس از ۴۰ گام به حدود ۹۷درصد رسیده است و بهبود داشته است. ضمنا اپتیمایزر مشابه همان مرحله قبل adam است. تصویر خروجی شبکه در این مورد نیز در بخش پیوست موجود است.

[11] Punctuation
[12] Stop
[13] Word tokenization
[14] Stemming
[15] Lemmatization

بخش چهارم

یکی از مسائل مهم در گرایش سیالات مهندسی مکانیک مساله‌ی تکمیل عکس‌های آزمایشگاهی PIV است. بواسطه پیچیدگی هندسه‌های موجود و تنوع مسائل، انجام آزمایشات، بسیار میتواند راه‌گشا باشد. آزمایش PIV از معروف‌ترین روش‌های تجربی برای سنجش میدان سرعت است. خروجی این آزمایش در مدت چند ثانیه میتواند چندین هزار عکس تولید کند. از مشکلات این عکس‌ها باید به وجود نقاط تعریف نشده در تصاویر اشاره کرد. در واقع با آشفته شدن جریان و میل به سمت میدان سرعت توربولانسی، تعداد نقاط تعریف شده بیشتر میشود. این نقاط تعریف نشده ممکن است به صورت نقطه‌ای پراکنده باشند یا به صورت خوشه‌ای وجود داشته باشند. استفاده از این عکس‌ها بدون تکمیل آنها ممکن نیست. لازم است پیش پردازش‌های لازم روی عکس‌ها صورت بگیرد سپس جاهای خالی با مقادیر مناسب تکمیل شوند.

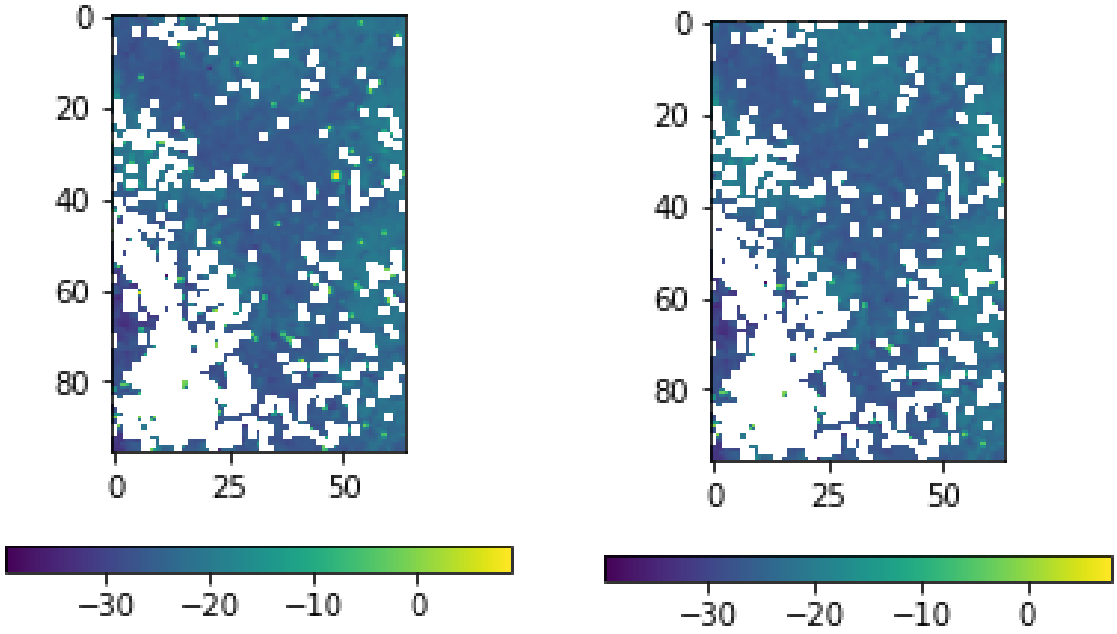
پیش پردازش های لازم برای انجام کار نسبتا گسترده و تا حدی وابسته به مراحل بعدی و کاربردی است که از شکل‌ها انتظار میرود. قالب کلی دادهPIV به صورت یک ماتریس چهار کاناله تبدیل میشود. کانال اول، شماره عکس را نشان میدهد. کانال دوم و سوم موقعیت هر پیکسل روی عکس را تعیین میکنند و کانال چهارم که دو حالت صفر و یک دارد مشخص میکند راستای سرعت افقی است یا عمودی. در ابتدا بررسی شده است که هر یک از عکس‌ها چند درصد داده تعریف شده دارند چرا که ممکن است یک عکس هیچ داده تعریف نشده‌ای نداشته باشد و یک عکس فقط ۴۰ درصد داده تعریف شده داشته باشد. هیستوگرام شکل ۱۳ برحسب تعداد عکس‌ها این مساله را در یک دیتاست نمونه مشخص می‌کند. در گام بعدی ماتریس چهار کاناله جدیدی تولید شده که از ماتریس اصلی کپی بگیرد تا تغییرات روی دیتای اصلی ایجاد نشود. سپس در میان عکس‌ها جستجو شده است. تعداد نقاط تعریف شده و تعریف نشده در هر عکس شمارش شده است. چنانچه حاصل تقسیم تعداد نقاط تعریف شده برکل نقاط(حاصل جمع تعریف شده و تعریف نشده) از عدد ۰/۸۵ بیشتر باشد این عکس به عنوان یک عکس با کیفیت حفظ میشود و مشخصات این اسنپ‌شات نگهداری میشود. اما چنانچه درصد داده تعریف شده از ۸۵ درصد کمتر باشد این عکس، بی‌کیفیت خوانده شده و حذف میشود. دلیل این امر آن است که هر چند روش‌های یادگیری ماشین میتواند تمام عکس‌ها را پر کند اما چنانچه درصد زیادی از عکس مشخص نباشد عملا ممکن است مفهوم فیزیک آن پدیده بعد از بازسازی عکس زیر سوال برود و تا حد زیادی ارضا نشود. بدین منظور یک حد آستانه تعریف شده تا عکس پس از بازسازی فاصله زیادی با فیزیک داستان نگیرد.



شکل ۱۳- هیستوگرام توزیع تعداد عکس‌های یک دیتاست نمونه برحسب درصد تعریف شدگی داده

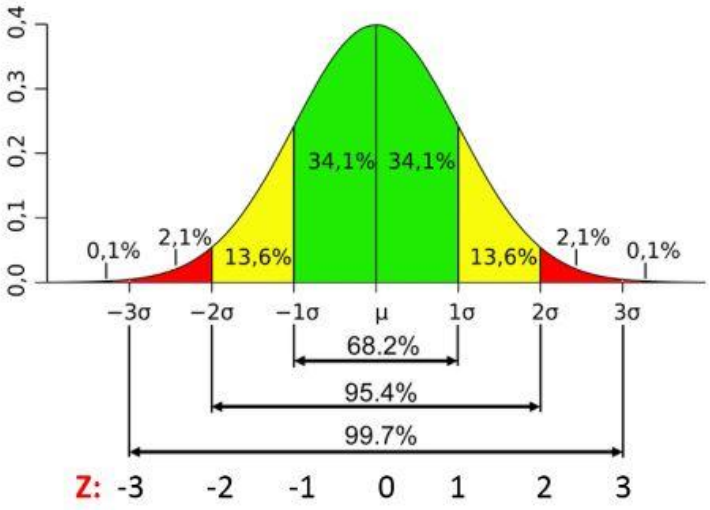
در این مرحله لازم است عکس‌ها از نظر وجود داده پرت بررسی شوند. در بعضی موارد ممکن است سرعت یک یا چند نقطه در هر عکس به طور صحیحی اندازه‌گیری نشده باشد. به طور بصری نیز بعضی از این نقاط معلوم هستند به طور مثال در شکل ۱۴ رنگ بعضی نقاط در دامنه، به طور واضحی با همسایگان‌شان متفاوت است. برای تشخیص اینکه آیا این نقاط داده پرت هستند یا خیر نیاز است یک معیار انتخاب شود تا بر اساس آن تصمیم‌گیری کرد. معیار انتخابی شش سیگما[۱۶] نام دارد که در داده‌هایی که احتمال میرود از توزیع نرمال پیروی میکنند استفاده میشود. مطابق شکل ۱۵، در این معیار یک محدوده معین برای سرعت‌ها لحاظ میشود چناچه سرعت در هر نقطه خارج از این محدوده باشد حذف می‌شود و با nan جایگزین خواهد شد.

[16] Six sigma



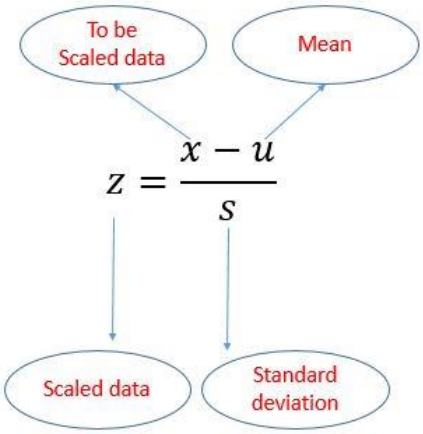
شکل۱۴- حذف داده پرت در یک اسنپ‌شات

سمت چپ: قبل از حذف داده پرت و سمت راست: بعد از حذف داده پرت

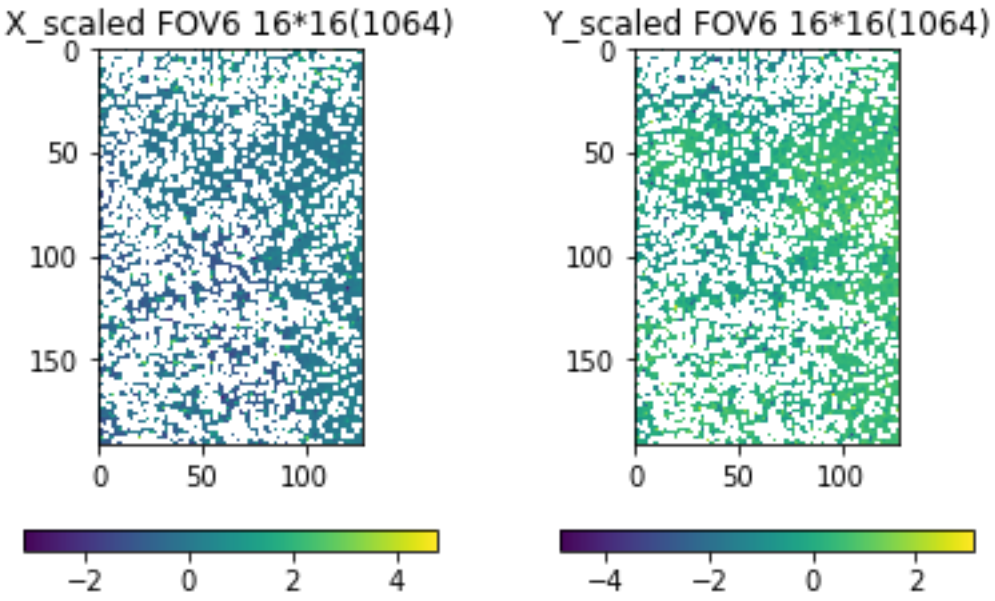


شکل۱۵- نمای کلی مفهوم روش شش سیگما

پس از حذف داده‌های پرت طبق الگوی انتخابی، لازم است که مقادیر سرعت نرمالایز[17] شوند. با توجه به سادگی این مفهوم از رابطه‌ای مطابق شکل۱۶ استفاده شده است و نهایتا تغییر مقادیر سرعت در شکل ۱۷ قابل مشاهده است.



شکل۱۶- شیوه نرمال‌کردن مقادیر سرعت



شکل۱۷- نمونه‌ای از اسنپ‌شات‌های نرمالایز شده

[17] Normalize

حال مراحل پیش پردازش تمام شده است و میتوان عملیات تکمیل کردن[18] را با استفاده از روش‌های یادگیری ماشین پیش برد.

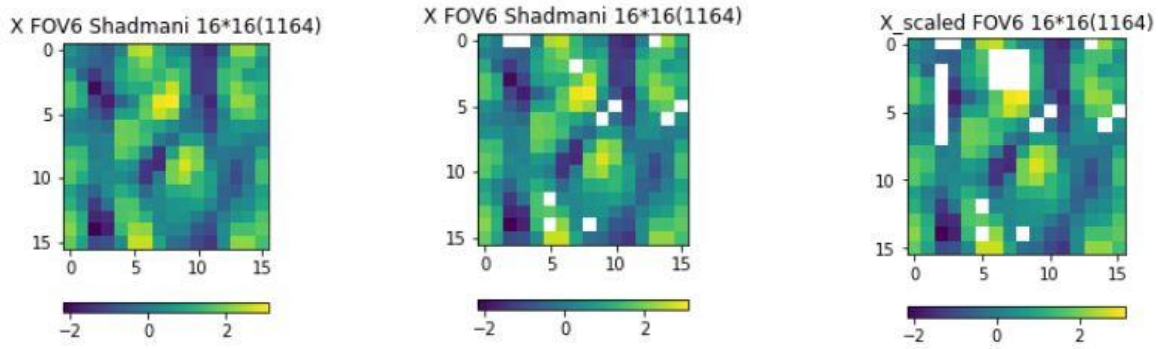
روش اول استفاده از الگوریتم KNNimputer میباشد. این ایمپیوتر به طور پیشفرض در کتابخانهsklearn موجود است. با انتخاب عدد ۴ برای تعداد همسایه‌ها میتوان اطمینان داشت که بر اساس همسایه‌های اطراف جای خالی پر خواهد شد و نتیجه در یک عکس در شکل ۱۹ قابل مشاهده است. البته این یک هایپر پارامتر است.

روش دوم استفاده از میانگین‌گیری است. بدین منظور میتوان در خانه‌خالی میانگین مقادیر تعریف شده در هر سطر از پیکسل‌ها را جایگزین کرد. البته این انتخاب میتوان بر اساس ستون نیز صورت بگیرد. یعنی میانگین‌گیری روی هر ستون انجام شود. راه دیگر آن است که میانگین گیری برای هر پیکسل خالی روی همان پیکسل و در میان عکس‌های متفاوت صورت بگیرد. یعنی مثلا اگر در موقعیت(x,y) مقدار سرعت تعریف‌نشده است روی تمام اسنپ‌شات‌هایی که این عدد به صورت تعریف‌شده وجود دارد میانگین‌گیری انجام شود که نتایج نشان داد علی‌رغم اینکه میانگین سطری و ستونی تفاوت چندانی با هم ندارند اما این روش آخر خروجی مناسبی ندارد. چرا که هر عکس در لحظه متفاوتی عکس‌برداری شده است و از لحاظ منطقی نمی‌توان توقع داشت مقدار سرعت یک پیکسل به اسنپ‌شات‌هایی در زمان‌های بعدی و قبلی مربوط باشد. خروجی این روش در شکل ۱۸موجود است.

در میان روش‌های ذکر شده روش میانگین گیری در سطر(یا ستون) گزینه مناسب‌تری بنظر میرسد که از لحاظ منطقی دقیق‌تر است و احتمالا خطای کمتری ایجاد میکند. حال برای آنکه بتوان سنجید که به طور دقیق کدام روش مناسب‌تر است نیاز است داده‌ای داشت که به عنوان داده تست بتوان از آن بهره برد. به بیان بهتر لازم است عکس‌هایی داشت که در آنها نقاط تعریف نشده وجود ندارد. سپس به طور مصنوعی مجموعه‌ای از نقاط تعریف نشده ایجاد کرد. پس از پایان بازسازی عکس‌های میتوان دقت را سنجید و بین روش‌های انتخابی مقایسه کرد. برای ایجاد نقاط تعریف نشده میتوان روش‌های مختلفی بکار برد. اساسا این کار بر اساس خلاقیت هر فرد میتواند متفاوت انجام شود. روشی که بکار گرفته شده انتخاب دو عدد تصادفی است. این دو عدد در محدوده تعداد پیکسل‌های راستای افقی و عمودی عکس‌ها است. این نقاط با nan جایگزین شده‌اند. این مرحله آنقدر تکرار شده تا تعداد نقاط خالی تولید شده مطابق عدد انتخابی باشد. مثلا چنانچه تولید ۲۰ نقطه مورد نظر باشد باید حداقل ۲۰ مرتبه این روند تکرار عدد تصادفی تولید شود. البته تجربه نشان داد لازم است بررسی صورت بگیرد آیا این نقطه خالی است یا خیر. یعنی در گام اول وقتی فقط عمل برای ۲۰مرتبه تکرار شد در بعضی موارد تعداد نقاط خالی شده کمتر از ۲۰ بود چون یک موقعیت بر حسب تصادف دوبار خالی شده بود. پس از نگاه کلی به ظاهر تصاویر مشاهده شد تفاوت فاحشی بین این عکس‌ها با شکل جاهای خالی اصلی وجود داشت. بدین منظور لازم

[18] Imputation

بنظر رسید تا تعدادی قسمت تعریف نشده خوشه‌ای به عکس‌ها اضافه شود. برای این هدف دو عدد تصادفی دیگر تولید شده تا موقعیت مورد نظر برای تولید خوشه مشخص شود. سپس دو عدد تصادفی دیگر تولید شده که از نصف تعداد پیکسل‌های هر راستا (افقی و عمودی) کمتر باشد. این قید از آن جهت لحاظ شده که احیانا عدد تصادفی آنچنان بزرگ نباشد که بیشتر یک سطر یا ستون به داده تعریف نشده تبدیل شود. سپس مستطیلی به تعداد پیکسل‌های دو عدد تصادفی دوم در موقعیت دو عدد تصادفی اول خالی شده و به nan تبدیل خواهد شد. حال پس از اعمال روش‌های بازسازی میتوان با داشتن مقادیر اصلی مقدار خطا را سنجید. عملا مساله به یادگیری با نظارت تبدیل شده است. ایده دیگر برای تولید جاهای خالی آن است که موقعیت نقاط تعریف نشده از روی یکی از عکس‌های اصلی که تعدادی جای خالی دارد شبیه‌سازی شود.



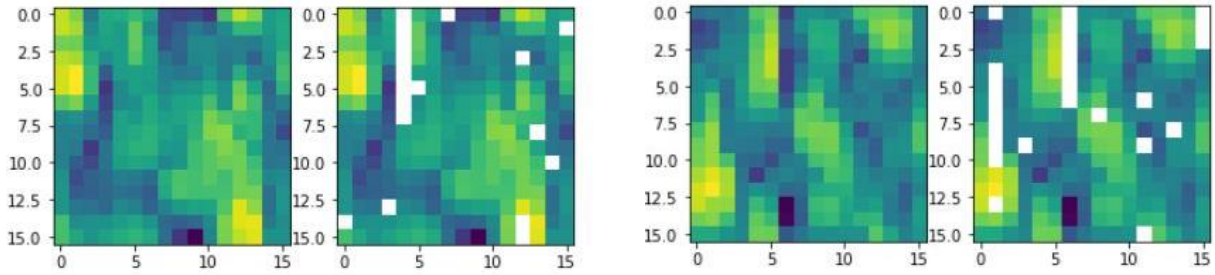
شکل ۱۸- نمایی از اعمال تغییرات روی داده DNS

به ترتیب از چپ به راست: تصویر اول: بدون اعمال تغییر تصویر میانی: پس از افزودن نقاط نقاط تهی تکی و تصویر آخر: پس از اضافه کردن نقاط خالی خوشه‌ای

جدول۹: مقایسه مقدار مطلق خطا در دو روش میانگین گیری و knn

روش میانگین‌گیری	روش KNN
۱۰/۵۸	۲۷/۵۹

مطابق جدول ۹ به طور واضح خطای حالت میانگین گیری کمتر است. ضمنا مقادیر خطا به صورت مطلق و بدون میانگین گیری گزارش شده تا اختلاف آن به وضوح قابل مشاهده باشد. ضمنا نمونه‌ای از تصاویر نیز در شکل ۱۹ برای مقایسه قابل مشاهده هستند.



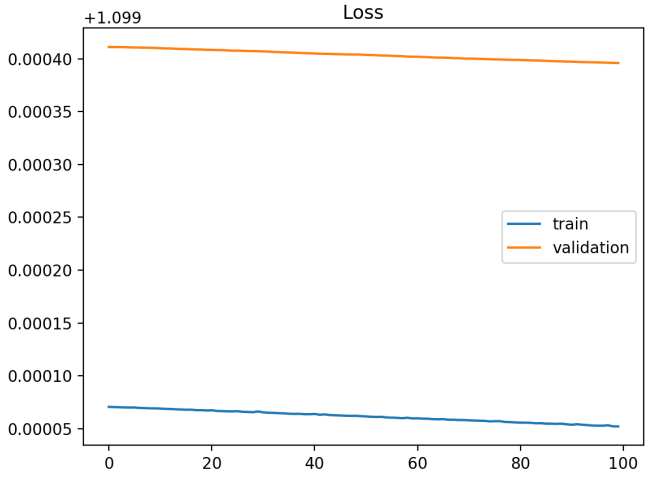
شکل۱۹- نمونه تصاویر تکمیل شده

سمت راست: KNN سمت چپ: میانگین‌گیری هر سطر

پیوست

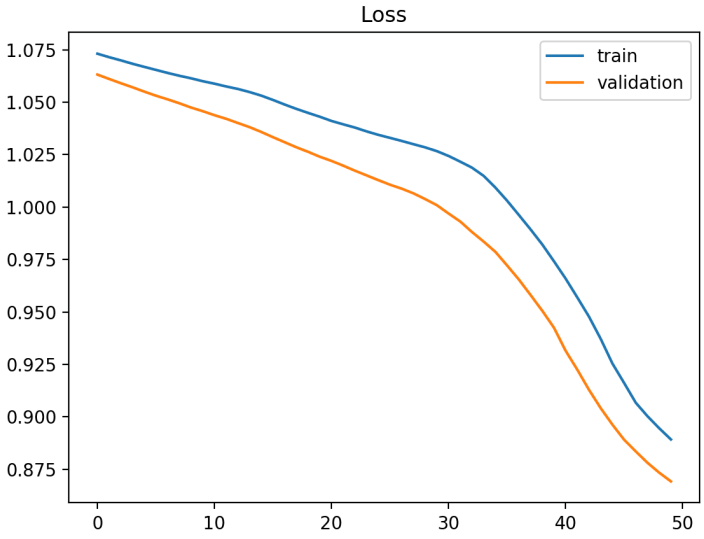
تحلیل نمودارهای accuracy-loss مربوط به بخش اول:

در مورد تفسیر نمودارهای accuracy-loss میتوان اشاره کرد که این نمودارها اصولا نمایان‌گر سه حالت میباشند. یا کم برازش [19] اتفاق افتاده و مدل آنچنان ساده است که نمیتواند الگوهای مساله موجود را یاد بگیرد. در این حالت یک خط صاف، نمایانگر منحنی یادگیری میتوان باشد. یا حتی ممکن است یک منحنی اکیدا نزولی از ابتدا تا انتها باشد که در انتها نیز همچنان نزولی است و به معنای آن است که امکان یادگیری بیشتر وجود دارد. ممکن است بیش‌برازش[20] اتفاق افتاده باشد، بدین معنا که عملا داده حفظ شده و همه چیز حتی نوسانات[21] رندم را نیز فراگرفته است. در این حالت مقدار ضرر در نمودار train همچنان کاهش می یابد و مقدار ضرر در نمودار تست تا حدی کم میشود و سپس شروع به افزایش دارد. حالت سوم آن است که مدل به خوبی فیت شده است[22]. در این حال مقدار ضرر برای تست و ترین در انتها از هم فاصله زیادی ندارند و هر دو روند کاهشی داشته‌اند تا به یک نقطه مشخص نزدیک شده‌اند. ادامه این مسیر میتواند به اورفیت شدن مدل میل کند.

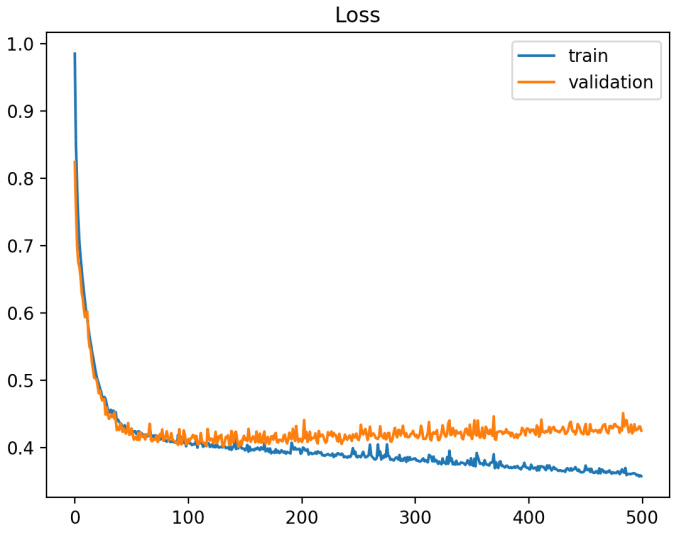


شکل الف- نمونه‌ای از شکل‌های کم برازش

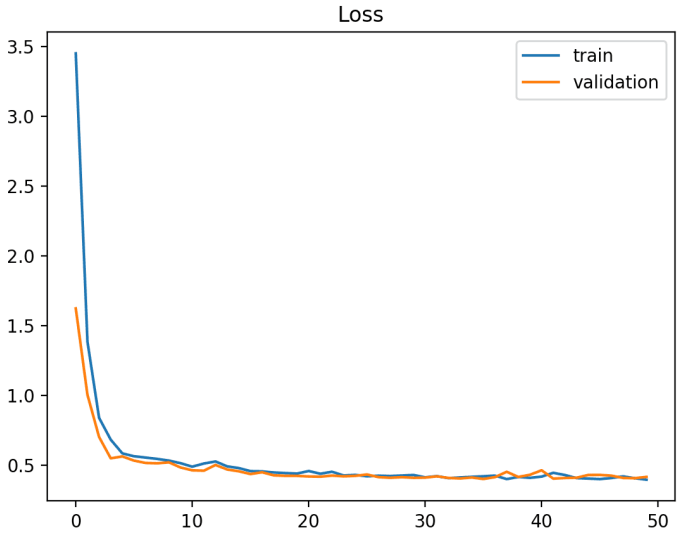
[19] Under fit
[20] Over fit
[21] fluctuation
[22] Good fit



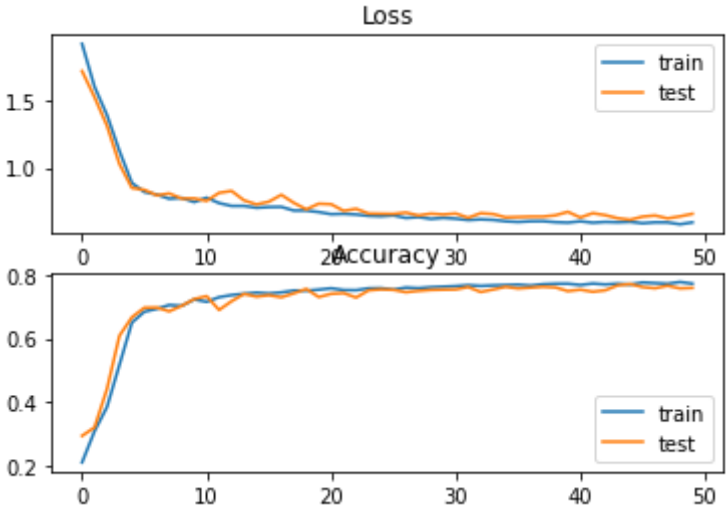
شکل ب- نمونه‌ای از شکل‌های کم برازش



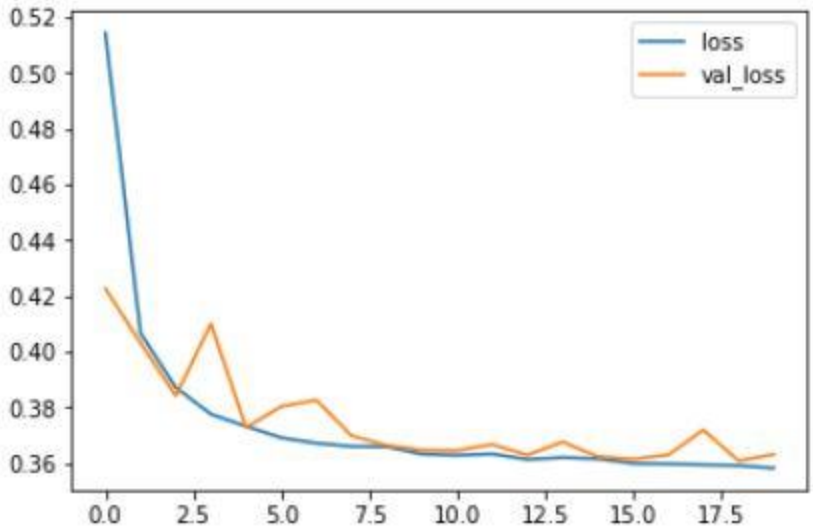
شکل پ- نمونه‌ای از شکل‌های بیش برازش



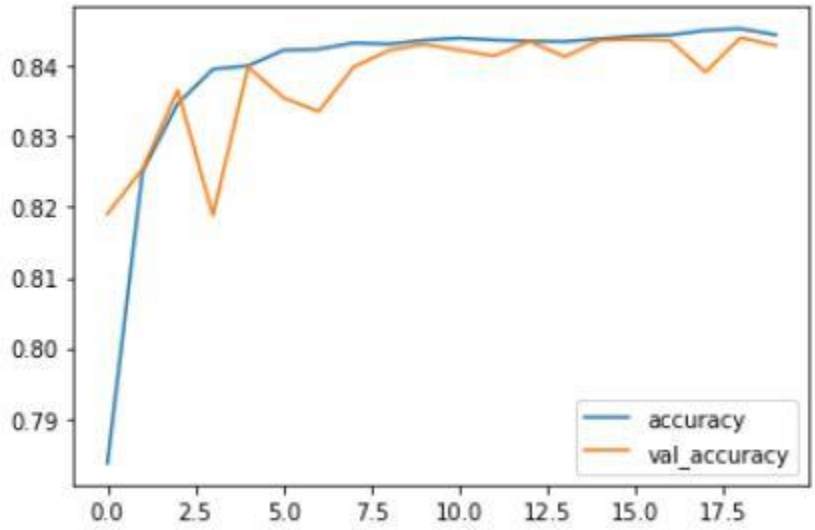
شکل ت- نمونه‌ای از شکل‌های فیت‌شدن مناسب



شکل ث- مدل انتخابی در بخش اول



شکل ج- مدل اعمال‌شده در بخش دوم



شکل چ- مدل اعمال‌شده در بخش دوم

```
Epoch 1/10
74/74 - 2s - loss: 1.4747 - accuracy: 0.3189 - 2s/epoch - 25ms/step
Epoch 2/10
74/74 - 1s - loss: 1.2489 - accuracy: 0.4980 - 1s/epoch - 14ms/step
Epoch 3/10
74/74 - 1s - loss: 1.0239 - accuracy: 0.6336 - 1s/epoch - 14ms/step
Epoch 4/10
74/74 - 1s - loss: 0.8329 - accuracy: 0.7111 - 1s/epoch - 14ms/step
Epoch 5/10
74/74 - 1s - loss: 0.6869 - accuracy: 0.7775 - 1s/epoch - 14ms/step
Epoch 6/10
74/74 - 1s - loss: 0.5727 - accuracy: 0.8233 - 1s/epoch - 14ms/step
Epoch 7/10
74/74 - 1s - loss: 0.4800 - accuracy: 0.8537 - 1s/epoch - 14ms/step
Epoch 8/10
74/74 - 1s - loss: 0.4030 - accuracy: 0.8791 - 1s/epoch - 14ms/step
Epoch 9/10
74/74 - 1s - loss: 0.3402 - accuracy: 0.9023 - 1s/epoch - 14ms/step
Epoch 10/10
74/74 - 1s - loss: 0.2896 - accuracy: 0.9185 - 1s/epoch - 14ms/step
```

شکل ح- خروجی شبکه عصبی بدون پیش پردازش متن

```
37/37 - 1s - loss: 0.0798 - accuracy: 0.9727 - 604ms/epoch - 16ms/step
Epoch 28/40
37/37 - 1s - loss: 0.0811 - accuracy: 0.9742 - 612ms/epoch - 17ms/step
Epoch 29/40
37/37 - 1s - loss: 0.0767 - accuracy: 0.9755 - 615ms/epoch - 17ms/step
Epoch 30/40
37/37 - 1s - loss: 0.0738 - accuracy: 0.9751 - 599ms/epoch - 16ms/step
Epoch 31/40
37/37 - 1s - loss: 0.0750 - accuracy: 0.9734 - 624ms/epoch - 17ms/step
Epoch 32/40
37/37 - 1s - loss: 0.0731 - accuracy: 0.9761 - 610ms/epoch - 16ms/step
Epoch 33/40
37/37 - 1s - loss: 0.0724 - accuracy: 0.9763 - 644ms/epoch - 17ms/step
Epoch 34/40
37/37 - 1s - loss: 0.0686 - accuracy: 0.9762 - 613ms/epoch - 17ms/step
Epoch 35/40
37/37 - 1s - loss: 0.0692 - accuracy: 0.9759 - 605ms/epoch - 16ms/step
Epoch 36/40
37/37 - 1s - loss: 0.0666 - accuracy: 0.9764 - 611ms/epoch - 17ms/step
Epoch 37/40
37/37 - 1s - loss: 0.0616 - accuracy: 0.9774 - 605ms/epoch - 16ms/step
Epoch 38/40
37/37 - 1s - loss: 0.0628 - accuracy: 0.9764 - 614ms/epoch - 17ms/step
Epoch 39/40
37/37 - 1s - loss: 0.0639 - accuracy: 0.9779 - 629ms/epoch - 17ms/step
```

شکل خ- خروجی شبکه عصبی بعد پیش پردازش متن