به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین امتیازی

نام عضو اول	فاطمه رشیدی شهری
شماره دانشجویی	۶۱۰۳۹۹۱۳۱
نام عضو دوم	آراد وزیرپناه
شماره دانشجویی	۶۱۰۳۹۹۱۸۲

فهرست

شکل‌ها

جدول‌ها

قوانین

قبل از پاسخ دادن به پرسش‌ها، موارد زیر را با دقت مطالعه نمایید:

- از پاسخ‌های خود یک گزارش در قالبی که در صفحه‌ی درس در سامانه‌ی Elearn با نام ***REPORTS_TEMPLATE.docx*** قرار داده شده تهیه نمایید.
- پیشنهاد می‌شود تمرین‌ها را در قالب گروه‌های دو نفره انجام دهید. (بیش از دو نفر مجاز نیست و تحویل تک نفره نیز نمره‌ی اضافی ندارد) توجه نمایید الزامی در یکسان ماندن اعضای گروه تا انتهای ترم وجود ندارد. (یعنی، می‌توانید تمرین اول را با شخص A و تمرین دوم را با شخص B و ... انجام دهید)
- **کیفیت گزارش شما در فرآیند تصحیح از اهمیت ویژه‌ای برخوردار است**؛ بنابراین، لطفا تمامی نکات و فرض‌هایی را که در پیاده‌سازی‌ها و محاسبات خود در نظر می‌گیرید در گزارش ذکر کنید.
- در گزارش خود مطابق با آنچه در قالب نمونه قرار داده شده، برای شکل‌ها زیرنویس و برای جدول‌ها بالانویس در نظر بگیرید.
- الزامی به ارائه توضیح جزئیات کد در گزارش نیست، اما باید نتایج بدست آمده از آن را گزارش و تحلیل کنید.
- **تحلیل نتایج الزامی می‌باشد، حتی اگر در صورت پرسش اشاره‌ای به آن نشده باشد.**
- **دستیاران آموزشی ملزم به اجرا کردن کدهای شما نیستند**؛ بنابراین، هرگونه نتیجه و یا تحلیلی که در صورت پرسش از شما خواسته شده را به طور واضح و کامل در گزارش بیاورید. در صورت عدم رعایت این مورد، بدیهی است که از نمره تمرین کسر می‌شود.
- **کدها حتما باید در قالب نوت‌بوک با پسوند .ipynb تهیه شوند، در پایان کار، تمامی کد اجرا شود و خروجی هر سلول حتما در این فایل ارسالی شما ذخیره شده باشد.** بنابراین برای مثال اگر خروجی سلولی یک نمودار است که در گزارش آورده‌اید، این نمودار باید هم در گزارش هم در نوت‌بوک کدها وجود داشته باشد.
- **در صورت مشاهده‌ی تقلب امتیاز تمامی افراد شرکت‌کننده در آن، 100- لحاظ می‌شود.**
- تنها زبان برنامه نویسی مجاز **Python** است.
- **استفاده از کدهای آماده برای تمرین‌ها به هیچ وجه مجاز نیست. در صورتی که دو گروه از یک منبع مشترک استفاده کنند و کدهای مشابه تحویل دهند، تقلب محسوب می‌شود.**

- نحوه محاسبه تاخیر به این شکل است: پس از پایان رسیدن مهلت ارسال گزارش، حداکثر تا یک هفته امکان ارسال با تاخیر وجود دارد، پس از این یک هفته نمره آن تکلیف برای شما صفر خواهد شد.
  - سه روز اول: بدون جریمه
  - روز چهارم: ۵ درصد
  - روز پنجم: ۱۰ درصد
  - روز ششم: ۱۵ درصد
  - روز هفتم: ۲۰ درصد
- حداکثر نمره‌ای که برای هر سوال می‌توان اخد کرد ۱۰۰ بوده و اگر مجموع بارم یک **سوال** بیشتر از ۱۰۰ باشد، در صورت اخد نمره بیشتر از ۱۰۰، اعمال نخواهد شد.
  - برای مثال: اگر نمره اخذ شده از سوال ۱ برابر ۱۰۵ و نمره سوال ۲ برابر ۹۵ باشد، نمره نهایی تمرین ۹۷.۵ خواهد بود و نه ۱۰۰.
- لطفا گزارش، کدها و سایر ضمایم را به در یک پوشه با نام زیر قرار داده و آن را فشرده سازید، سپس در سامانه‌ی Elearn بارگذاری نمایید:

HW[Number] _[Lastname]_[StudentNumber]_[Lastname]_[StudentNumber].zip

(مثال: HW1_Ahmadi_810199101_Bagheri_810199102.zip)

- برای گروه‌های دو نفره، بارگذاری تمرین از جانب یکی از اعضا کافی است ولی پیشنهاد می‌شود هر دو نفر بارگذاری نمایند.

پرسش ۱. تولید برچسب به کمک خوشه‌بندی

۱-۱. دادگان

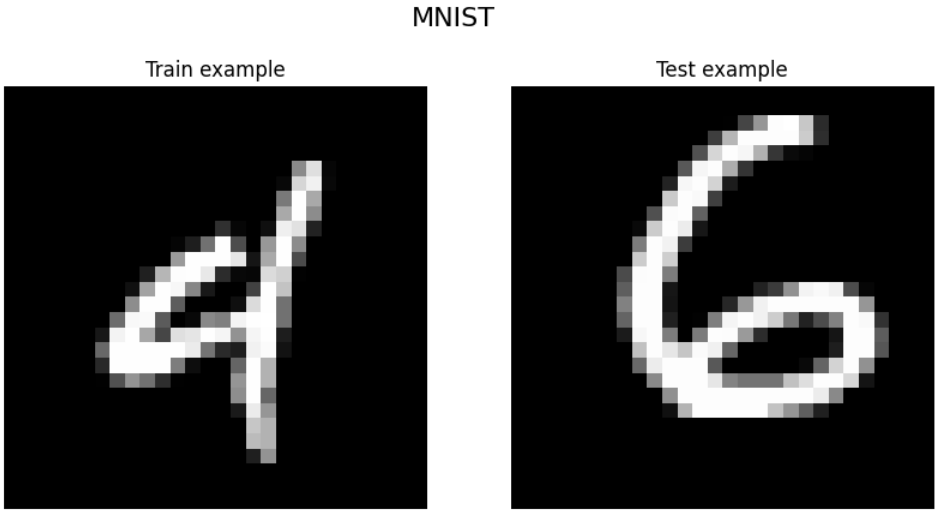
۱-۱-۱. MNIST

این دیتاست شامل تصاویری از اعداد دست‌نویس ۰ تا ۹ و همچنین برچسب متناظر با آن‌ها است. در ابتدا آن‌ها را به صورت داده‌های آموزش و تست لود کرده و سپس تصاویر را نرمالایز می‌کنیم. این کار را با تقسیم کردن هر پیکسل بر ۲۵۵ انجام می‌دهیم. اطلاعات مربوط به shape و تعداد تصاویر به صورت زیر است:

- در کل ۶۰۰۰۰ تصویر به عنوان داده آموزش و ۱۰۰۰۰ تصویر به عنوان داده تست در این دیتاست وجود دارد.
- تصاویر موجود در این دیتاست، به صورت سیاه-سفید بوده (تنها یک بعد برای عمق تصویر وجود دارد) و اندازه آن‌ها به صورت ۲۸×۲۸ می‌باشد.

در نهایت ۲۵ درصد از داده‌های آموزش را به عنوان دادگان اعتبارسنجی در نظر می‌گیریم، بنابراین در کل تعداد دادگان در هر مجموعه به این صورت خواهد بود:

- آموزش: ۴۵۰۰۰
- اعتبارسنجی: ۱۵۰۰۰
- تست: ۱۰۰۰۰



شکل 1. نمونه‌ای از مجموعه تست و آموزش **MNIST**

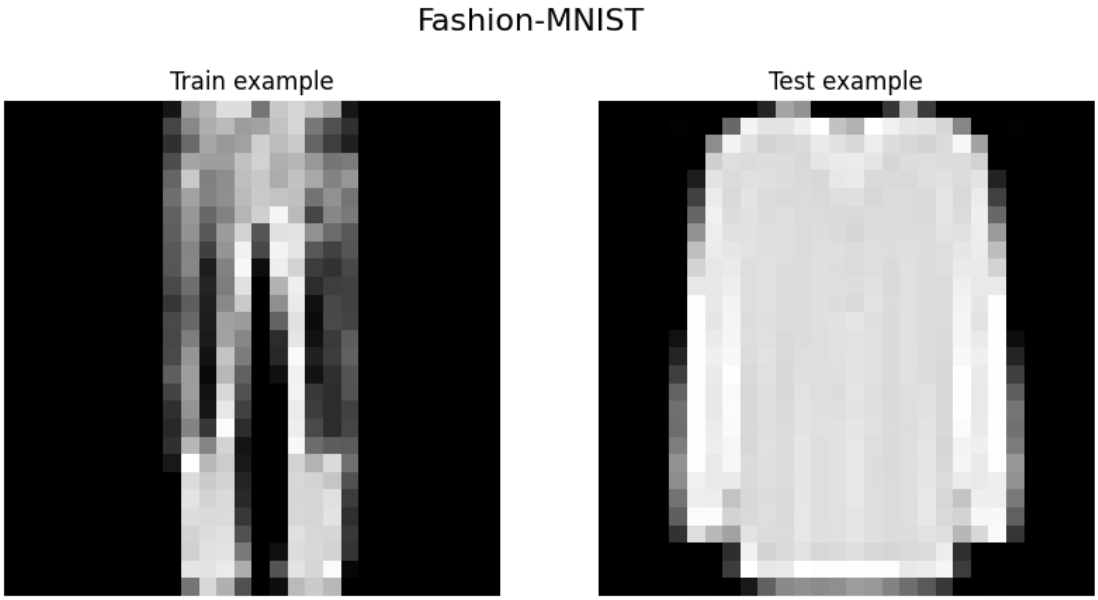
۱-۱-۲. Fashion MNIST

در این دیتاست تصاویر مربوط به انواع پوشاک (۱۰ نوع پوشاک) و همچنین برچسب متناظر با آن‌ها موجود است. مانند دادگان MNIST، این دادگان را نیز نرمالایز می‌کنیم. (این تصاویر نیز سیاه-سفید هستند)

- در کل ۶۰۰۰۰ تصویر به عنوان داده آموزش و ۱۰۰۰۰ تصویر به عنوان داده تست در این دیتاست وجود دارد.
- تصاویر موجود در این دیتاست، به صورت سیاه-سفید بوده (تنها یک بعد برای عمق تصویر وجود دارد) و اندازه آن‌ها به صورت ۲۸×۲۸ می‌باشد.

در نهایت ۲۵ درصد از داده‌های آموزش را به عنوان دادگان اعتبارسنجی در نظر می‌گیریم، بنابراین در کل تعداد دادگان در هر مجموعه به این صورت خواهد بود:

- آموزش: ۴۵۰۰۰
- اعتبارسنجی: ۱۵۰۰۰
- تست: ۱۰۰۰۰



شکل 2. نمونه‌ای از مجموعه تست و آموزش **Fashion MNIST**

۱-۲. شبکه مورد استفاده

۱-۲-۱. بخش Encoder

این بخش از مدل به عنوان ورودی یک تصویر ۲۸×۲۸ گرفته و با استفاده از لایه‌های کانولوشنی و dropout و BatchNormalizationهای متوالی، سعی می‌کند feature mapهای کوچک‌تری بدست آورند (در حالی که تعداد آن‌ها افزایش میابد). در نهایت feature mapهای بدست آمده را flat می‌کنیم و سپس با استفاده از یک لایه fully connected، شامل تعداد مورد نظر نورون برای تشکیل فضای latent (در اینجا از ۶ نورون استفاده کردیم)، فضای latent را بدست می‌آوریم که بعدا می‌توانیم از این فضا و مکان نمونه‌های موجود، clustering انجام دهیم و نمونه‌ها را برچسب گذاری کنیم. summary آن به صورت زیر است:

Model: "Encoder"

Layer (type)	Output Shape	Param #
input_layer_9 (InputLayer)	(None, 28, 28, 1)	0
conv2d_12 (Conv2D)	(None, 28, 28, 16)	160
max_pooling2d_12 (MaxPooling2D)	(None, 14, 14, 16)	0
dropout_24 (Dropout)	(None, 14, 14, 16)	0
batch_normalization_32 (BatchNormalization)	(None, 14, 14, 16)	64
conv2d_13 (Conv2D)	(None, 14, 14, 32)	4,640
max_pooling2d_13 (MaxPooling2D)	(None, 7, 7, 32)	0
dropout_25 (Dropout)	(None, 7, 7, 32)	0
batch_normalization_33 (BatchNormalization)	(None, 7, 7, 32)	128
conv2d_14 (Conv2D)	(None, 7, 7, 64)	18,496
max_pooling2d_14 (MaxPooling2D)	(None, 4, 4, 64)	0
dropout_26 (Dropout)	(None, 4, 4, 64)	0
batch_normalization_34 (BatchNormalization)	(None, 4, 4, 64)	256
conv2d_15 (Conv2D)	(None, 4, 4, 128)	73,856
max_pooling2d_15 (MaxPooling2D)	(None, 2, 2, 128)	0
dropout_27 (Dropout)	(None, 2, 2, 128)	0
batch_normalization_35 (BatchNormalization)	(None, 2, 2, 128)	512
flatten_3 (Flatten)	(None, 512)	0
dense_6 (Dense)	(None, 6)	3,078

Total params: 101,190 (395.27 KB)
Trainable params: 100,710 (393.40 KB)
Non-trainable params: 480 (1.88 KB)

۱-۲-۲. بخش Decoder

در این بخش ابتدا با استفاده از لایه‌های fully connected، ابتدا ورودی‌های با طول ۶ را دریافت کرده و سپس دوباره در لایه بعدی از FC با ۵۱۲ نورون (مانند بخش انکودر بعد از flat کردن) استفاده کرده و ابعاد را بزرگ‌تر می‌کنیم. بعد از آن از لایه‌های Conv2DTranspose، ZeroPadding، DropOut و BatchNormalization استفاده می‌کنیم تا در نهایت دوباره یک تصویر سیاه-سفید به اندازه ۲۸×۲۸ در خروجی داشته باشیم. summary آن به صورت زیر است:

Model: "Decoder"

Layer (type)	Output Shape	Param #
input_layer_10 (InputLayer)	(None, 6)	0
dense_7 (Dense)	(None, 512)	3,584
reshape_3 (Reshape)	(None, 2, 2, 128)	0
batch_normalization_36 (BatchNormalization)	(None, 2, 2, 128)	512
conv2d_transpose_27 (Conv2DTranspose)	(None, 2, 2, 128)	147,584
zero_padding2d_12 (ZeroPadding2D)	(None, 4, 4, 128)	0
conv2d_transpose_28 (Conv2DTranspose)	(None, 8, 8, 128)	147,584
dropout_28 (Dropout)	(None, 8, 8, 128)	0
batch_normalization_37 (BatchNormalization)	(None, 8, 8, 128)	512
conv2d_transpose_29 (Conv2DTranspose)	(None, 8, 8, 64)	73,792
zero_padding2d_13 (ZeroPadding2D)	(None, 10, 10, 64)	0
conv2d_transpose_30 (Conv2DTranspose)	(None, 20, 20, 64)	36,928
dropout_29 (Dropout)	(None, 20, 20, 64)	0
batch_normalization_38 (BatchNormalization)	(None, 20, 20, 64)	256
conv2d_transpose_31 (Conv2DTranspose)	(None, 20, 20, 32)	18,464
zero_padding2d_14 (ZeroPadding2D)	(None, 24, 24, 32)	0
conv2d_transpose_32	(None, 24, 24, 32)	9,248

(Conv2DTranspose)		
dropout_30 (Dropout)	(None, 24, 24, 32)	0
batch_normalization_39 (BatchNormalization)	(None, 24, 24, 32)	128
conv2d_transpose_33 (Conv2DTranspose)	(None, 24, 24, 16)	4,624
zero_padding2d_15 (ZeroPadding2D)	(None, 28, 28, 16)	0
conv2d_transpose_34 (Conv2DTranspose)	(None, 28, 28, 16)	2,320
dropout_31 (Dropout)	(None, 28, 28, 16)	0
batch_normalization_40 (BatchNormalization)	(None, 28, 28, 16)	64
conv2d_transpose_35 (Conv2DTranspose)	(None, 28, 28, 1)	145
batch_normalization_41 (BatchNormalization)	(None, 28, 28, 1)	4

Total params: 445,749 (1.70 MB)
Trainable params: 445,011 (1.70 MB)
Non-trainable params: 738 (2.88 KB)
Model: "mnist_autoencoder"

۱-۲-۳. مدل اصلی

در نهایت بخش انکودر و دیکودر را با هم ترکیب کرده به صورتی که ابتدا ورودی وارد انکودر شده و سپس وارد دیکودر شود. بنابراین این مدل Convolutional AutoEncoder ما را تشکیل داده و می‌توانیم برای تسک مورد نظر از آن استفاده کنیم. summary آن به این صورت است:

Model: "autoencoder"

Layer (type)	Output Shape	Param #
input_layer_11 (InputLayer)	(None, 28, 28, 1)	0
Encoder (Functional)	(None, 6)	101,190
Decoder (Functional)	(None, 28, 28, 1)	445,749

Total params: 546,939 (2.09 MB)
Trainable params: 545,721 (2.08 MB)
Non-trainable params: 1,218 (4.76 KB)

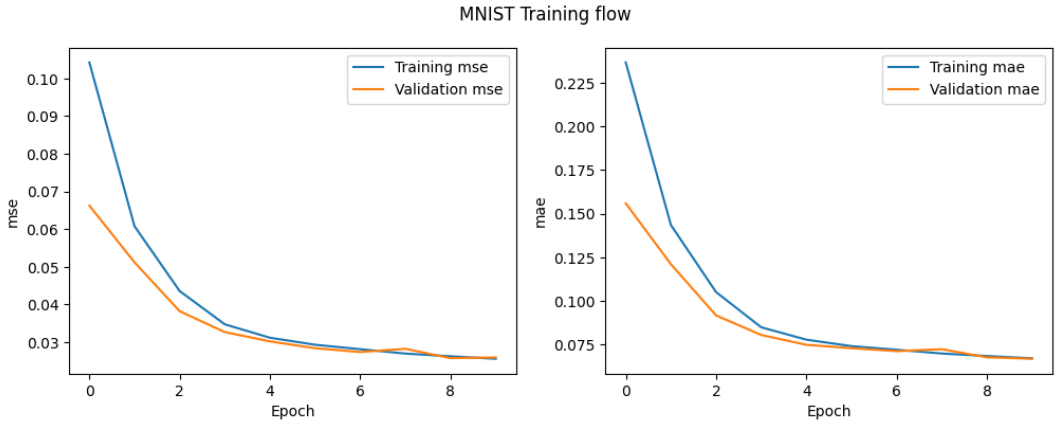
همان‌طور که قابل مشاهده است، مدل تقارن داشته و در ابتدا تصاویر را به فضای ۶ بعدی برده و دوباره یک تصویر ۲۸×۲۸ از این فضا می‌سازد.

۱-۳. آموزش مدل

نتایج استفاده از این مدل برای هر کدام از داده‌ها را بررسی می‌کنیم.

۱-۳-۱. آموزش مدل با MNIST

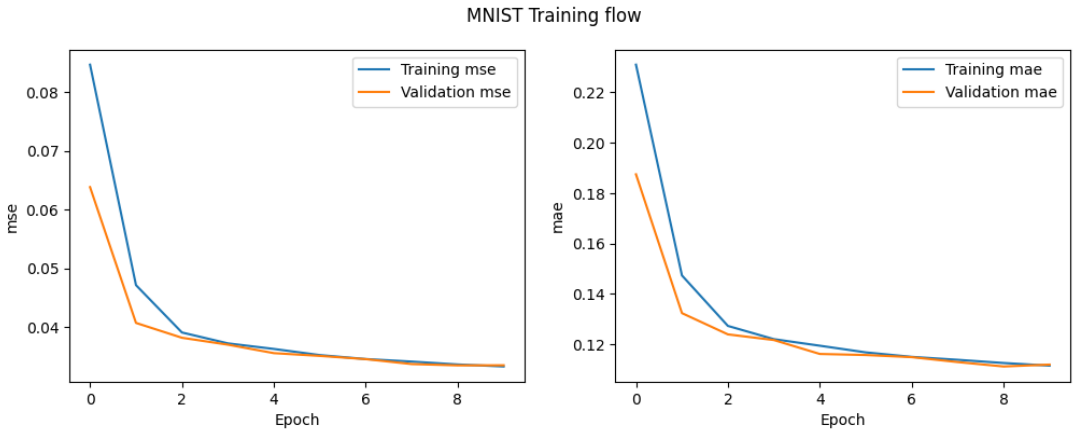
با استفاده از پارامترهای گفته شده در صورت سوال، مدل را با داده‌های MNIST، آموزش می‌دهیم. برای تابع loss مدل از MSE استفاده می‌کنیم. نتایج آن به صورت زیر خواهد بود.



شکل 3. نتایج آموزش مدل با **MNIST**

۱-۳-۲. آموزش مدل با Fashion MNIST

با استفاده از پارامترهای گفته شده در صورت سوال، مدل را با داده‌های Fashion MNIST، آموزش می‌دهیم. برای تابع loss مدل از MSE استفاده می‌کنیم. نتایج آن به صورت زیر خواهد بود.



شکل 4. نتایج آموزش مدل با **Fashion MNIST**

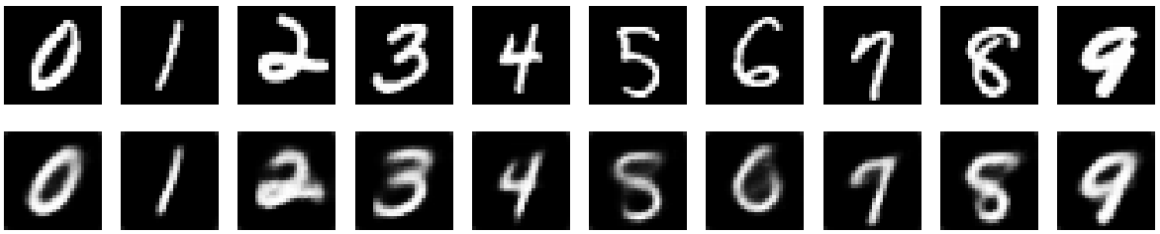
همانطور که مشاهده می‌کنیم، مدل‌ها به خوبی آموزش دیده و mse و mae را به ازای تصاویر ورودی و خروجی برای هر مدل به اندازه کافی کوچک شده.

۱-۴. ارزیابی مدل‌ها و مشاهده خروجی آن‌ها

در این بخش نتیجه مدل بر روی داده تست هر کدام از داده‌ها را بررسی کرده و همچنین یک نمونه تصادفی از تصاویر و خروجی آن‌ها از مدل را نمایش می‌دهیم.

۱-۴-۱. مدل MNIST

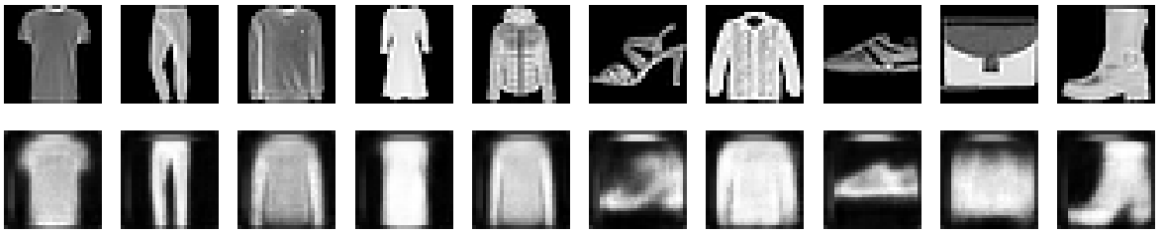
نتایج مدل روی داده‌های تست به صورت `mae: 0.0667 mse: 0.0258` است که حتی نتیجه بهتری از آن‌چه که روی داده‌های آموزش و اعتبارسنجی بدست آمده دارد.



شکل 5. نمونه ورودی و خروجی برای مدل **MNIST**

۱-۴-۲. مدل Fashion MNIST

نتایج مدل روی داده‌های تست به صورت `mae: 0.1120 mse: 0.0334` است که حتی نتیجه بهتری از آن‌چه که روی داده‌های آموزش و اعتبارسنجی بدست آمده دارد.



شکل 6. نمونه ورودی و خروجی برای مدل **Fashion MNIST**

۱-۵. خوشه بندی

Silhouette Score معیاری برای سنجش میزان شباهت یک نمونه به خوشه خود (انسجام) در مقایسه با خوشه‌های دیگر (جداسازی) است. این امتیاز بین -۱ و ۱ قرار دارد به طوری که:

- امتیازی نزدیک به ۱ نشان می‌دهد که نمونه فاصله زیادی از خوشه‌های همسایه دارد.
- امتیاز ۱ نشان می‌دهد که نمونه بر روی مرز تصمیم بین دو خوشه همسایه قرار دارد یا بسیار نزدیک به آن است.
- امتیازی نزدیک به -۱ نشان می‌دهد که ممکن است نمونه به خوشه نادرست اختصاص داده شده باشد.

برای محاسبه امتیاز سیلوئت برای نتیجه یک خوشه‌بندی، به این صورت عمل می‌کنیم:

1. محاسبه میانگین فاصله درون‌خوشه‌ای (a): برای هر نمونه، میانگین فاصله تا تمام نقاط دیگر در همان خوشه را محاسبه کنید.
2. محاسبه میانگین فاصله نزدیکترین خوشه دیگر (b): برای هر نمونه، میانگین فاصله تا تمام نقاط در نزدیک‌ترین خوشه‌ای که نمونه در آن قرار ندارد را محاسبه کنید.
3. محاسبه امتیاز سیلوئت برای هر نمونه: امتیاز سیلوئت برای یک نمونه به صورت زیر محاسبه می‌شود:

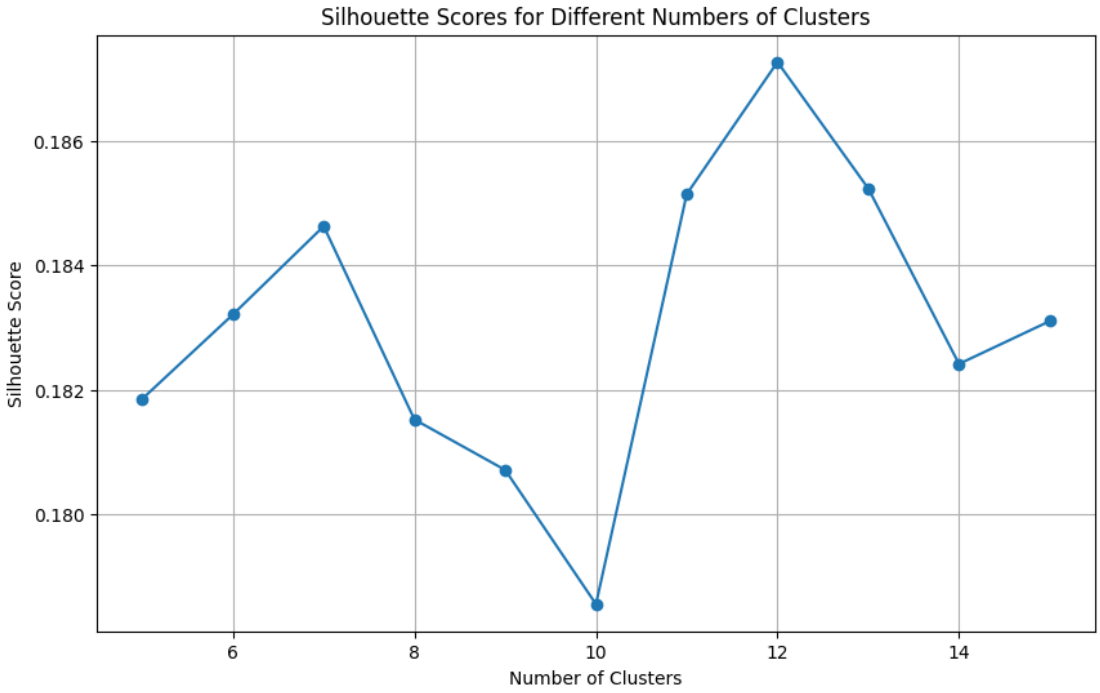
$$s(i) = \frac{b(i) - a(i)}{\max\big(a(i), b(i)\big)}$$

4. **میانگین امتیازات سیلوئت**: امتیاز کلی سیلوئت برای تمام نمونه‌ها میانگین امتیازات سیلوئت تمام نمونه‌ها است.

حال به کمک الگوریتم KMeans، همه داده‌های یک مجموعه (متشکل از داده‌های آموزش، تست و اعتبارسنجی) را با استفاده از نمایش آن‌ها در فضای latent که با استفاده از بخش انکورد autoencoder تعریف شده بدست آمده، خوشه‌بندی می‌کنیم. این کار را با بهترین تعداد خوشه‌ای که بالاترین Silhouette Score را دارد انجام داده و خوشه‌ها را مشخص می‌کنیم.

۱-۵-۱. خوشه‌بندی MNIST

نمودار Silhouette Score آن به صورت زیر است. بنابراین بهترین تعداد خوشه برای این مجموعه، ۱۲ است که تا حدودی به تعداد کلاس‌های واقعی نزدیک است. احتمالا دلیل اینکه تعداد کلاس‌های بعد از خوشه‌بندی بیشتر از تعداد کلاس‌های واقعی است، این می‌باشد که مدل تمایز زیادی بین بعضی از اعضای برخی از کلاس‌ها میگذارد که باعث تشکیل کلاس‌های جدید می‌شود. در واقع می‌توان گفت مدل تا حد کمی، overfit کرده.



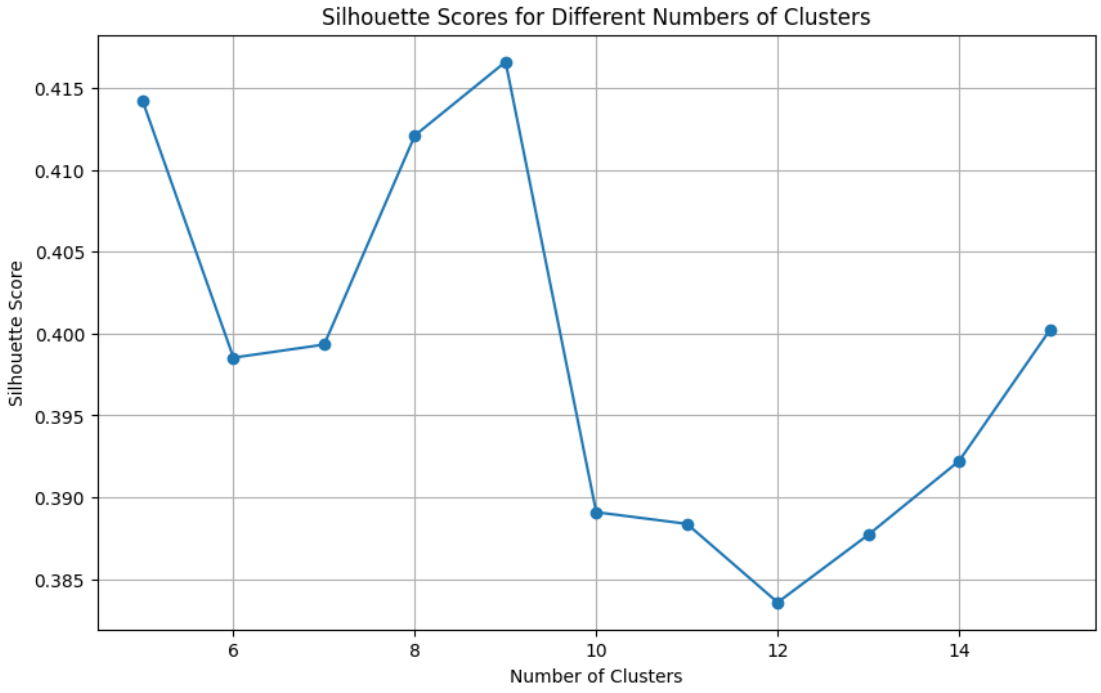
شکل 7. نمودار **Silhouette Score** برای **MNIST**

همچنین نتیجه خوشه‌بندی ۱۲ کلاسه آن را می‌توان در نمودار میله‌ای شکل ۹ دید که نمایاگر این است که هر خوشه که توسط الگوریتم KMeans تشخیص داده شده، در حقیقت شامل چه کلاس‌هایی است داده اصلی است.

برای مثال خوشه ۰ و ۲ بیشتر شامل اعداد ۳ و ۸ می‌باشند، چرا که در زبان انگلیسی، این دو عدد تا حدی نزدیک به یکدیگر می‌باشند. خوشه ۳ و ۶، عدد ۱ را تشخیص داده‌اند. خوشه ۱۰، عندد ۲ و خوشه ۷ عدد ۰ و خوشه ۹ و ۵ عدد ۶ را نشان می‌دهند.

۱-۵-۲. خوشه‌بندی Fashion MNIST

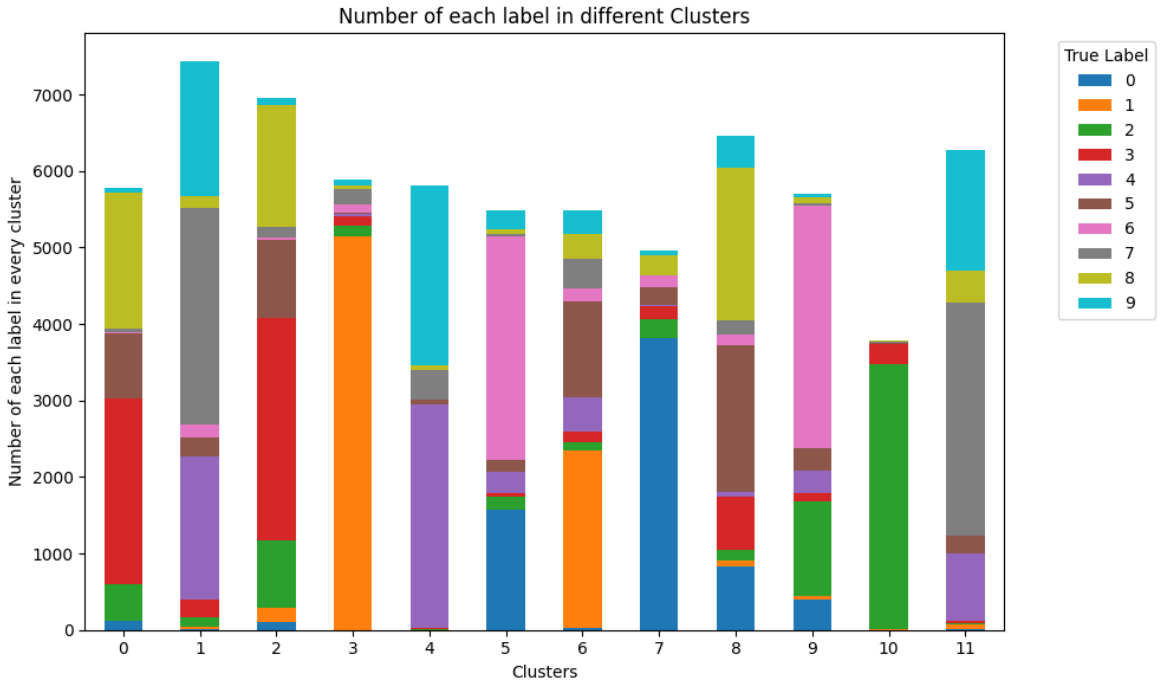
نمودار Silhouette Score آن به صورت زیر است. بنابراین بهترین تعداد خوشه برای این مجموعه، ۹ است که تا حد بسیار خوبی به تعداد کلاس‌های واقعی نزدیک است. احتمالا دلیل اینکه تعداد کلاس‌های بعد از خوشه‌بندی کم‌تر از تعداد کلاس‌های واقعی است، این می‌باشد که مدل تمایز کمی بین بعضی از کلاس‌ها میگذارد که باعث حذف یک کلاس می‌شود.



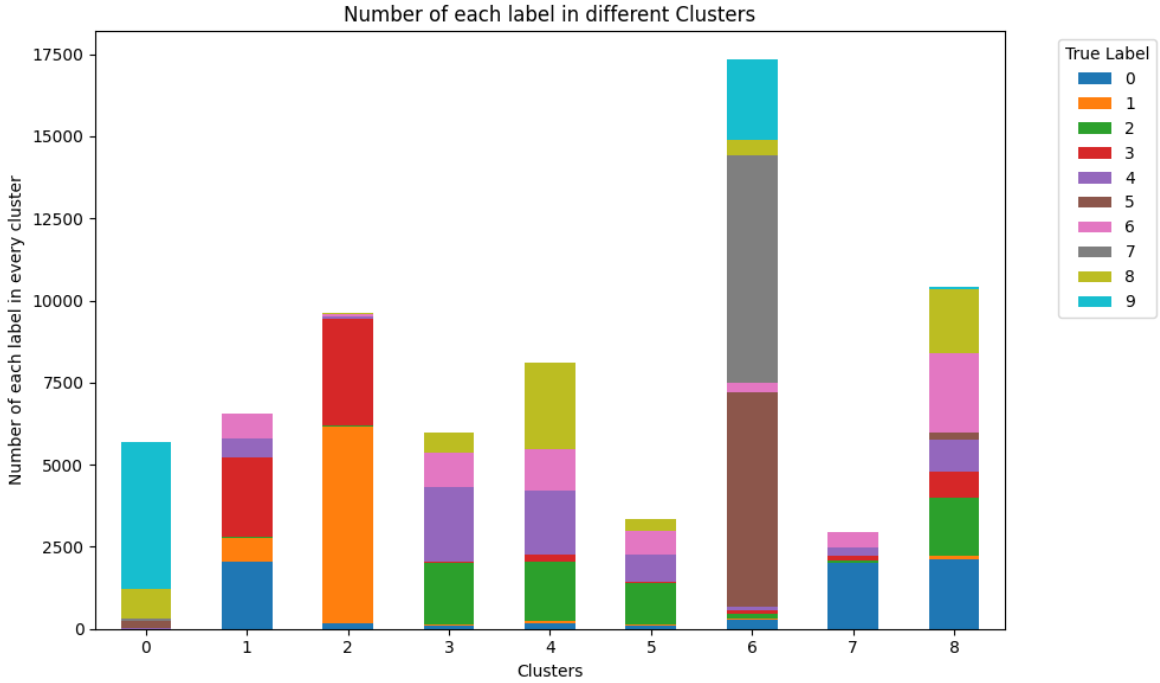
شکل 8. نمودار **Silhouette Score** برای **Fashion MNIST**

همچنین نتیجه خوشه‌بندی ۱۲ کلاسه آن را می‌توان در نمودار میله‌ای شکل ۱۰ دید که نمایاگر این است که هر خوشه که توسط الگوریتم KMeans تشخیص داده شده، در حقیقت شامل چه کلاس‌هایی است داده اصلی است.

همانگونه که انتظار داشتیم، کلاس ۵ و ۷ با هم ترکیب شده و یک خوشه را تشکیل داده‌اند که دلیل کم شدن تعداد خوشه‌ها نیز همین است. کلاس‌های ۲، ۴ و ۶ تقریبا بین ۴ کلاس تقسیم شده و احتمالا شکل و شمایل نزدیک به همی دارند که در یک خوشه جمع شده‌اند. و مابقی کلاس‌ها معمولا در یک خوشه هستند و آن خوشه بیانگر همان کلاس خواهد بود.



شکل 9. خوشه‌بندی **MNIST**



شکل 10.خوشه‌بندی **Fashion MNIST**

به طور مشخص مدل عملکرد بهتری روی داده‌های Fashion MNIST داشته چرا که mapping بهتری به فضای latent داشته و تعداد خوشه‌ها نزدیک‌تر به تعداد واقعی کلاس‌ها است. دلیل این احتمالا جداپذیرتر بودن دادگان Fashion MNIST نسبت به خود MNIST است.

پرسش ۲. افزایش داده در مدل FaBert

۲-۱. Data augmentation در NLP

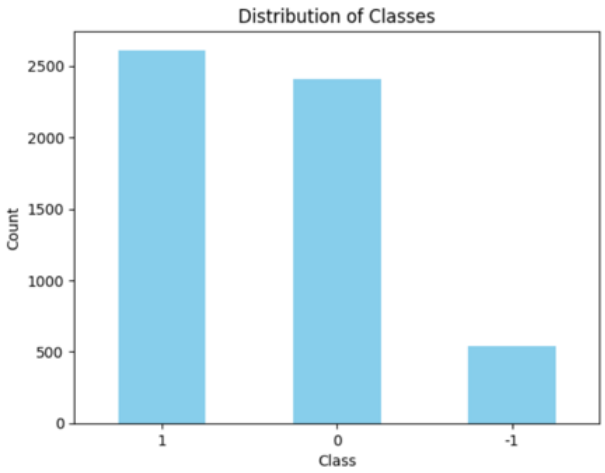
برای افزایش دادگان در پردازش زبان‌های طبیعی، راهکار های زیر قابل انجام است:

- Synonym replacement: در این روش، کلمات متن، با کلمات مترادف خود جایگزین می‌شوند و جملات ساختاری متفاوتی را می‌سازند که منجر به افزایش اندازه‌ی مجموعه‌ی دادگان می‌شود.
- Random insertion: این روش به صورت تصادفی کلمات را از یک مجموعه‌ی از پیش تعیین شده به جملات اضافه می‌کند.
- Random swap: در این روش مکان کلمات در یک جمله به صورت تصادفی با هم جابجا می‌شود.
- Random deletion: در این روش، به صورت تصادفی کلماتی از جمله‌ها حذف می‌شوند.
- Noise injection: در این روش، نویزها تحت عنوان کارکترهای تصادفی و یا کلمات misspelled به جمله‌ها افزوده می‌شوند.
- Back translation: در این روش، جملات به یک زبان دیگر ترجمه شده و جمله‌ی ترجمه شده مجدد به زبان اولیه ترجمه می‌شود. از آنجا که الزاما جمله‌ی ترجمه شده به زبان اولیه همان جمله‌ی اول نیست، منجر به افزایش اندازه‌ی مجموعه دادگان می‌شود.
- Text generation with GANs: در این روش یا استفاده از مدل‌های generative جملات جدید ساخته می‌شود و تحت عنوان داده‌های تصنعی نزدیک به واقعیت، به مجموعه دادگان افزوده می‌شود.

حال، به توضیح بیشتر back translation می‌پردازیم. در این روش ساختار جملات ممکن است دچار تحول شوند اما معنای جملات معمولا ثابت باقی می‌ماند. به این ترتیب منجر می‌شود مدل در فرایند یادگیری از کلمات و ساختارهای جدید نیز یاد بگیرد. همچنین، چون برای افزایش دادگان بکار می‌رود، منجر می‌شود که مدل robustness بیشتری داشته باشد و از لحاظ generality نیز اولویت داشته باشد. همچنین، برای بکار گیری این متد، از هر دو جفت زبانی می‌توان بهره برد. از این رو، روشی مطلوب برای افزایش دادگان بشمار می‌رود.

۲-۲. پیش‌پردازش دادگان

با تغییر برچسب‌ها در مجموعه دادگان آموزشی طبق صورت پروژه و تبدیل آنها به ۳ کلاس -۱، ۰ و ۱، توزیع کلاسها به صورت زیر است:



شکل 11. توزیع‌کلاسها پس از مرج کردن برچسب‌ها

طبق نمودار فوق واضح است که بیشترین کامنت‌ها در دادگان آموزشی از کلاس ۱ هستند، یعنی برچسب آنها در مجموعه دادگان اصلی، ۱ یا ۲ بوده است. همچنین، نزدیک به ۲۵۰۰ کامنت از کلاس ۰ بوده اند و تعداد کمی از کامنت‌ها از کلاس -۱ بوده اند. این مسئله نشان می‌دهد که بخش کمی از مشتری‌ها طبق این دادگان از خرید خود رضایت نداشته‌اند.

پیش از پیش‌پردازش، تعداد توکن‌ها در هر کامنت را محاسبه می‌کنیم و در دیتافریم ضبط میکنیم. با جمع بستن این مقادیر متوجه میشویم که در مجموع و پیش از پیش‌پردازش، ۱۳۰۱۵۰ عدد توکن در این دیتاست آموزشی داریم.

برای پیش‌پردازش دادگان ابتدا جملات را نرمال میکنیم، یعنی متن ها را به متن فارسی نرمال تبدیل می‌کنیم؛ مثلا اعراب‌ها را حذف می‌کنیم و فاصله‌گذاری‌ها را اصلاح می‌کنیم. سپس آنها را tokenize می‌کنیم و زبان هر توکن را بررسی می‌کنیم، اگر فارسی بود، ریشه‌ی آن را برمیگردانیم و اگر فارسی نبود یا punctuation بود، آن را کلا در نظر نمیگیریم. این مراحل را روی هر دو دسته‌ی آموزشی و آزمایشی انجام می‌دهیم.

سپس، تعداد توکن‌ها در هر کامنت را مجددا پس از اعمال پردازش‌ها محاسبه می‌کنیم و در سطر متناظر با آن، این مقدار را اضافه می‌کنیم. در این ستون جدید، مجموعا ۳۹۸۱۰ توکن در مجموعه دادگان پردازش شده داریم.

همچنین، داده‌های آموزشی را به دو دسته‌ی آموزشی و اعتبارسنجی تقسیم می‌کنیم، طوریکه ۲۰٪ آنها را به صورت تصادفی به عنوان داده‌ی اعتبارسنجی در نظر میگیریم. در این صورت، تعداد دادگان به صورت زیر خواهد بود:

Train set	Validation set
4448	1113

جدول 1. تعداد داده‌های پردازش شده در هر یک از مجموعه‌های اعتبارسنجی و آموزشی

۳-۲. افزایش دادگان به روش back translation

در این بخش پس از اعمال پردازش‌های لازم روی داده‌ها، آنها را به روش back translation زیاد میکنیم. تعداد دادگان پس از آگمنت برابر ۸۸۹۶ است که دقیقا دوبرابر تعداد دادگان اصلی آموزشی است.

لازم به ذکر است فقط دادگان آموزشی را آگمنت می‌کنیم و داده‌های آزمایشی و اعتبارسنجی نیازی به آگمنتیشن ندارند. پس این مجموعه‌ها ثابت باقی می‌مانند و تنها مجموعه‌ی آموزشی دچار تغییر می‌شود.

تعدادی از داده‌ها قبل و بعد از اعمال back translation به صورت زیر است:

متن پردازش شده پس از اعمال back translation	متن پردازش شده - اصلی
نکته درست این است که هدفون موثرتر است	نکته درسته گوش گوشی تره ب بازده داره
اما نوار تاریک قابل مشاهده نیست	دست‌ک باند تاریک روشن دیده_نمی‌شود
لمس آن آسان است، درصد ریزش تبلت دیگری نیز وجود ...	براحت دس میشه درصد افتادن تبلت دیگه هست
پردازنده مرکزی توسط مایکروسافت به طور غیر ر ...	پردازنده مرکز پردازنده مرکز کنسول ۳۶۰ نا رس ...
صفحه نمایش گوشی 4 3 اینچ	صفحه نما گوش صفحه نما ۴ ۳ اینچ
صد آیفون به نماینده اس 5 اجازه نمی دهد، اجازه ...	صد صد گوش آیفون نمی‌تواند اجازه معرف اس پنج نم ...
اوم در گوش	اهم گوش فرو
عرض 18 میلی متر 2 0 معمولی 35 میلی متر 1 4 60 ...	واید ۱۸ میلی ۲ ۰ نرمال ۳۵ میلی‌متر ۱ ۴ ۶۰ میلی...
دکمه کنترل دوربین را تا کنید	چین دکمه کنترل دوربین
بگذر	بگذر

جدول 2. تعدادی از نمونه‌های اولیه و معادل افزوده شده‌ی آن به دیتاست

با ارزیابی دادگان افزوده شده میتوان نتیجه گرفت که مفهوم کلی جملات در حالت کلی حفظ شده‌اند اما کلمات عمدتا تغییر یافته‌اند و همان جملات قبلی را نداریم بلکه اغلب از مترادفات کلمات بهره گرفته شده است. لازم به ذکر است در صورتی که اگر در پیش‌پردازش تنها به نرمال کردن جملات بسنده می‌کردیم، آنگاه کیفیت ترجمه‌ی جملات احتمالا بهتر بود چراکه ترجمه‌ی یک جمله با در دست داشتن کل جمله‌ی مبدا بهتر رقم می‌خورد.

۴-۲. تنظیم دقیق مدل FaBert

برای هر دو مدل که قرار است روی دادگان اصلی و دادگان آگمنت شده یادگیری کنند، هایپرپارامترهای زیر را در نظر میگیریم:

Learning rate	epochs	Loss	Optimizer	Batch size
0.00001	5	Cross Entropy	Adamw	16

جدول 3. هایپرپارامترهای مورد استفاده در مدل‌های **FaBERT**

برای تنظیم دقیق این مدل روی دادگان پردازش شده، برای اینکه overfitting رخ ندهد، علاوه بر کلسیفایر، آخرین لایه‌ی مدل را نیز آن‌فریز کردیم و سایر لایه‌ها non-trainable باقی می‌مانند. در این صورت، تعداد کل پارامترهای قابل آموزش برابر ۷۰۹۰۱۷۹ عدد خواهد بود.

نتایج برای این مدل روی مجموعه دادگان اصلی آموزشی، اعتبارسنجی و آزمایشی به شرح زیر است:

Train accuracy	Train loss	Validation accuracy	Validation loss	Test accuracy	Test loss
59.51%	0.8420	58.13%	0.8841	59.01%	0.8834

جدول 4. نتایج آموزش مدل روی داده‌های آگمنت نشده

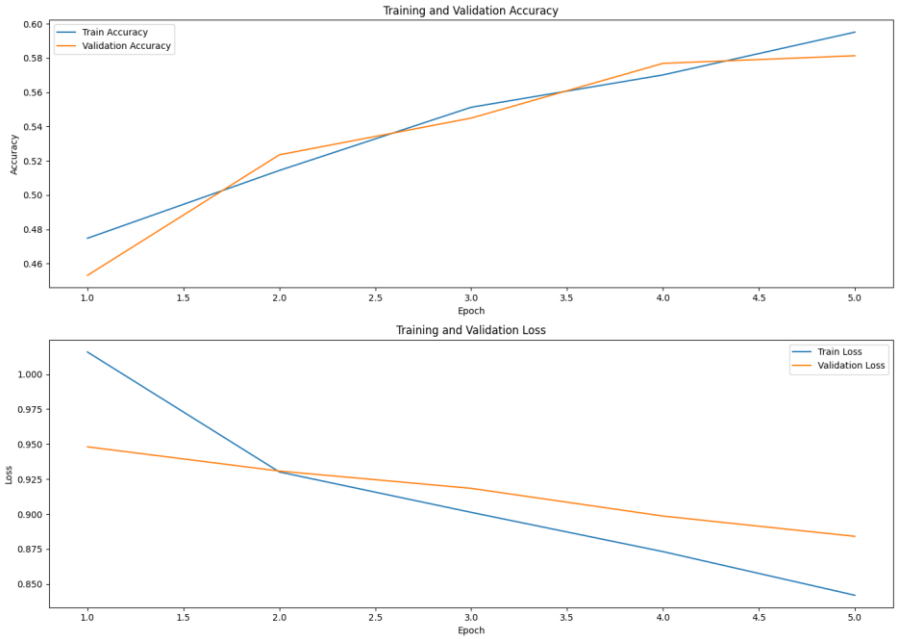
پس از آموزش مدل روی دادگان افزوده شده نیز نتایج زیر بدست می‌آید:

Train accuracy	Train loss	Validation accuracy	Validation loss	Test accuracy	Test loss
44.41%	1.0954	42.37%	1.0970	47.24%	1.0932

جدول 5. نتایج آموزش مدل روی داده‌های آگمنت شده

۵-۲. ارزیابی و تحلیل نتایج

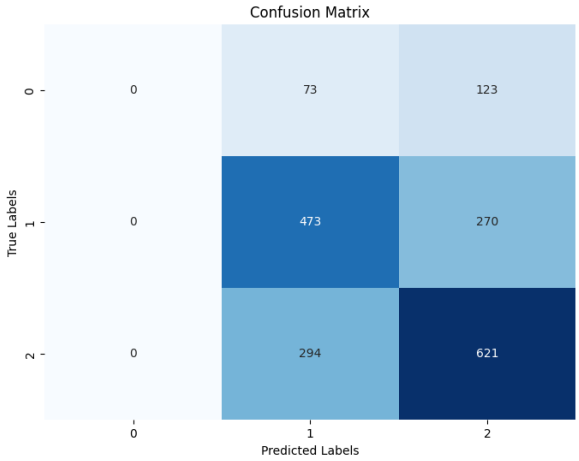
با آموزش مدل روی داده‌های آگمنت نشده با هایپرپارامترهای گفته شده در پنج ایپاک نتایج زیر بدست می‌آید:



شکل 12. نمودارهای دقت و هزینه برای مدل روی داده‌های اصلی در ۵ ایپاک

همانطور که در نمودارهای فوق مشخص است، در طی ۵ ایپاک، دقت و هزینه در هر دو مجموعه دادگان آموزشی و اعتبارسنجی به ترتیب در حال افزایش و کاهش است. این مسئله نوید بخش خوبی است. همچنین، اختلاف نمودارها برای دادگان آموزشی و اعتبارسنجی به قدری کم است که نمیتوان آن را برازش داده درنظر گرفت. نکته‌ی قابل توجه در تحلیل نمودارهای فوق این است که هزینه در دادگان اعتبارسنجی به صورت خطی و با شیب کمتری از دادگان آموزشی در حال کاهش است. همچنین، پیش بینی می‌شود با آموزش مدل طی ایپاک های بیشتر، عملکرد آن بهتر شود چراکه هیچ یک از نمودارها هنوز همگرا نشده اند و مدل همچنان پتانسیل یادگیری دارد.

ماتریس آشتفتگی نیز به صورت زیر است. در این ماتریس، لیبل متناظر با هر یک از کلاس های -۱، ۰ و ۱ به ترتیب ۰، ۱ و ۲ در نظر گرفته شده‌اند چراکه برای آموزش باید برچسپ های اعداد غیر منفی می‌بودند.



شکل 13. ماتریس آشفتگی متناظر با عملکرد مدل روی دادگان اصلی

از ماتریس فوق میتوان نتایج زیر را استنتاج کرد:

- از کلاس -۱ که عدم رضایت مشتری ها از خریدشان بود، هیچ کامنتی به درستی طبقه بندی نشده، بلکه ۷۳ نمونه از کلاس ۰ و ۱۲۳ مورد از کلاس ۱ به اشتباه پیش بینی شده اند.
- از کلاس دوم، کلاس ۰، تنها ۴۷۳ کامنت به درستی طبقه بندی شده اند و ۲۷۰ نمونه از این کلاس به اشتباه از کلاس ۱ پیش بینی شده اند.
- از کلاس ۱ که رضایت مشتریان از خریدشان است، تنها ۶۲۱ مورد به درستی طبقه بندی شده‌اند و ۲۹۴ مورد به اشتباه از کلاس ۰ پیش بینی شده اند.

F1-score نیز در این مدل پس از آموزش برابر ۰.۵۵۷۳ است که نمایانگر عملکرد چندان خوبی از شبکه نیست و نشان می‌دهد که تقریبا نیمی از نمونه ها به درستی پیش بینی شده اند.

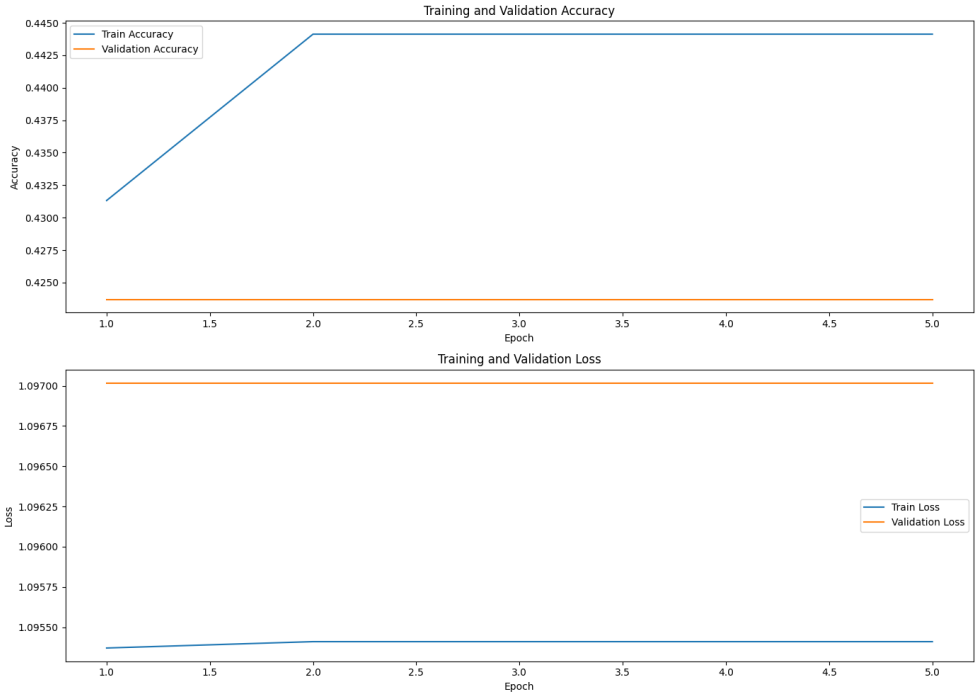
چند نمونه از دادگان ارزیابی که مدل به اشتباه پیش بینی کرده به صورت زیر هستند:

```
True Label: 0, Predicted Label: 2
Text: چیزا نمیتونه باشه

True Label: 2, Predicted Label: 1
Text: موارد نامیده دستگاه هواو شرایط فراه نموده_اس آن خواهد_نمود

True Label: 2, Predicted Label: 1
Text: می‌دان گوش ۶ پشتیبان سیس ریس شرک سامسونگ معرف نو ۲ دربار دستنیافتن ارزو دار روز گوشی سامسونگ استفاده

True Label: 1, Predicted Label: 2
Text: دوربین نمایشگر ۳ اینچ قرار رزولوشن ۲۳۰۰۰۰ نقطه ارائه

True Label: 1, Predicted Label: 2
Text: اجرا باز گرافیک داشته_باش
```

شکل 14. نمونه‌ای از دادگان به اشتباه پیش‌بینی شده توسط مدل روی دادگان اصلی

حال به آموزش مدل رو دادگان افزوده شده می‌پردازیم.

نمودارهای خواسته شده برای آموزش مدل روی دادگان افزوده شده به صورت زیر است:



شکل 15. نمودارهای دقت و هزینه برای مدل روی داده‌های افزوده شده در ۵ ایپاک

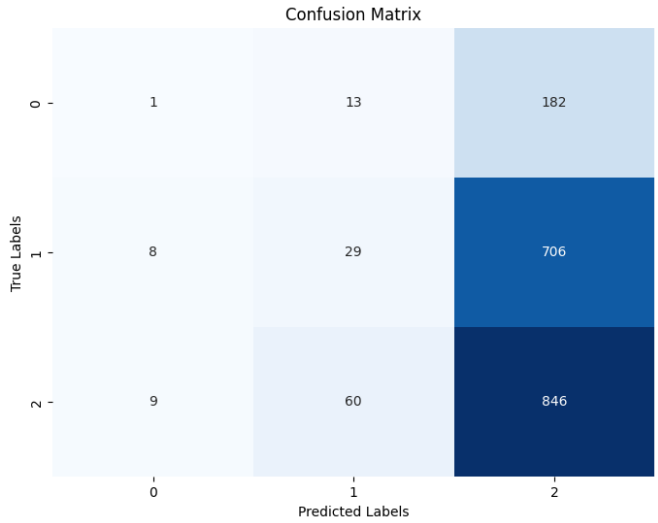
نمودار های فوق نشانگر عملکرد ضعیف مدل روی دادگان افزوده شده هستند. برای تحلیل این نمودارها چند سناریو وجود دارد:

ممکن است کیفیت دادگان آموزشی و اعتبارسنجی یکسان نباشد. این مسئله کاملا منطقی است چراکه دادگان اعتبارسنجی ما، همان دادگان پردازش شده‌ی اصلی هستند درحالیکه تنها نیمی دادگان آموزشی، این شرایط را دارند و مابقی، داده‌های افزوده شده هستند که کیفیت دادگان اصلی را ندارد.

سناریوی دوم، معماری مدل است. اگر مثلا یک لایه بیشتر آن‌فریز کنیم، احتمالا عملکرد مدل بهتر خواهد بود و روی دادگان دیده نشده عملکرد بهتری خواهد داشت.

همچنین ممکن است نرخ یادگیری خیلی بزرگ یا خیلی کوچک انتخاب شده باشد که با تغییر آن، مدل عملکرد بهتری داشته باشد.

ماتریس آشفتگی نیز به صورت زیر است:



شکل 16. ماتریس آشفتگی متناظر با عملکرد مدل روی دادگان افزوده شده

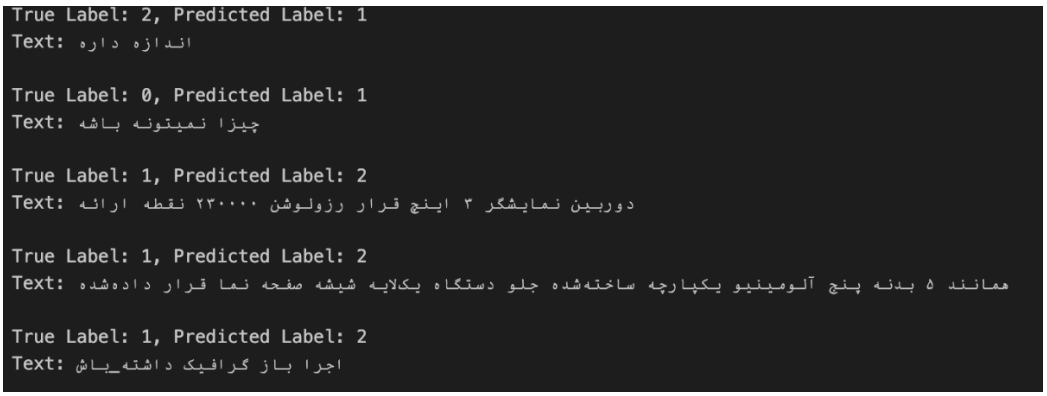
از ماتریس فوق میتوان موارد زیر را استنتاج کرد:

- تنها ۱ مورد از کلاس -۱ به درستی پیش بینی شده. ۱۳ مورد از این کلاس، از کلاس ۰ و ۱۸۲ مورد از این کلاس، به اشتباه از کلاس ۱ پیش‌بینی شده اند.
- تنها ۲۹ مورد از کلاس ۰ به درستی پیش بینی شده. ۸ مورد از این کلاس، از کلاس -۱ و ۷۰۶ مورد از این کلاس، به اشتباه از کلاس ۱ پیش‌بینی شده اند.
- ۸۴۶ مورد از کلاس ۱ به درستی پیش بینی شده. ۹ مورد از این کلاس، از کلاس -۱ و ۶۰ مورد از این کلاس، به اشتباه از کلاس ۰ پیش‌بینی شده اند.

F1-score وزن دار برای این حالت نیز برابر ۰.۳۴۳۷ است که نمایانگر این است که تنها حدود یک سوم از دادگان به درستی پیش‌بینی شده اند.

با تحلیل نتایج فوق میتوان دریافت مدل روی این دادگان اغلب کلاس ۱ را پیش بینی کرده است. از آنجا که تعداد دادگان مربوط به این کلاس نیز بیشتر از سایرین است، تا حدی این رفتار طبیعی است. برای رفع این مشکل باید یک دیتاست یکنواخت‌تر آماده کنیم که در آن، تعداد نمونه‌ها در کلاس‌های مختلف تقریبا یکسان است. در این صورت میتوان مطمئن بود مدل عملکرد بهتری ارائه خواهد داد.

همچنین، تعدادی از دادگان ارزیابی که به اشتباه پیش بینی شده اند، در شکل زیر آمده است:

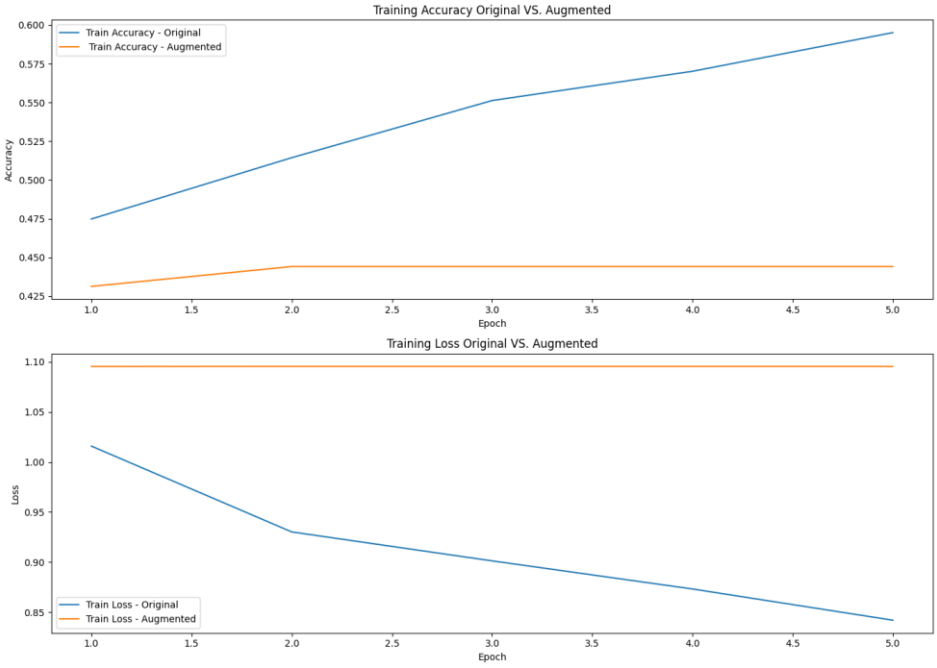


شکل 17. نمونه‌ای از دادگان به اشتباه پیش‌بینی شده توسط مدل روی دادگان افزوده شده

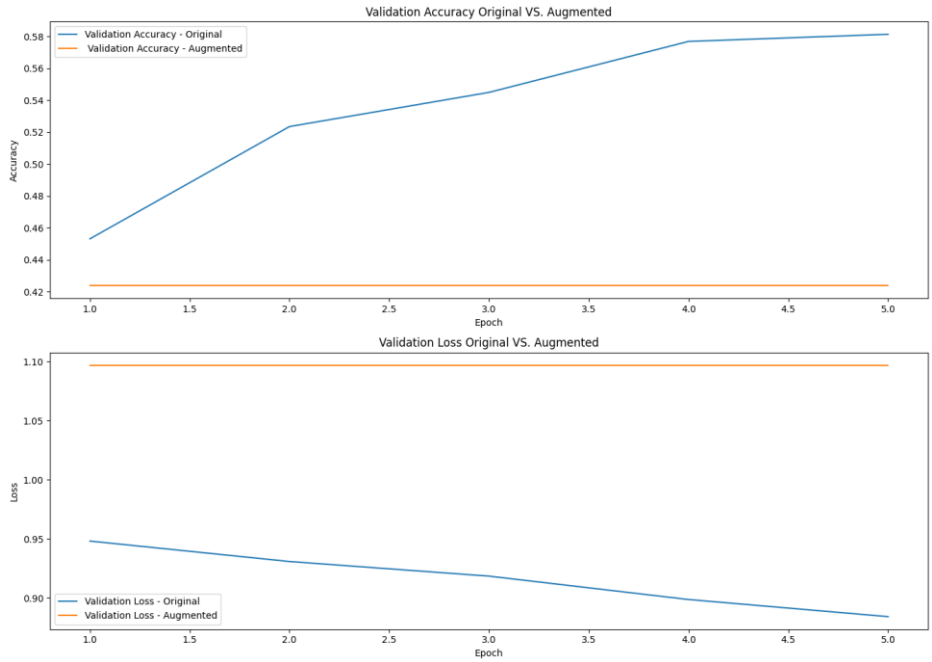
نتایج بدست آمده از بالا را میتوان به این صورت خلاصه کرد: همانطور که از جداول ۴ و ۵ قابل استنتاج است، مدل روی دادگان اصلی عملکرد بهتری دارد و تعداد بیشتری از نمونه ها را درست پیش‌بینی میکند که این برخلاف انتظار ماست. انتظار داشتیم پس از افزایش مجموعه دادگان، عملکرد مدل با همان معماری و پارامترها بهتر شود، اما نشد. این مسئله می‌تواند دلایل متنوعی داشته باشد که بنظر من مهم‌ترین آنها تفاوت دادگان آموزشی و اعتبارسنجی است. از آنجا که دادگان اعتبارسنجی همان دادگان اصلی پردازش شده بودند اما تنها نیمی از دادگان آموزشی واجد این شرایط بودند و سایر آنها دادگان افزوده شده بودند، لذا نمیتوان انتظار چندان خوبی از مدل داشت چراکه کیفیت ترجمه‌ها نیز چندان خوب نیست و داده‌ای که پس از پردازش، ترجمه و بازترجمه شود اندکی از کیفیت آن کاسته می‌شود. از جمله دلایل دیگر نیز میتوان به معماری مدل و هایپرپارامترها مثل نرخ یادگیری اشاره کرد.

همچنین، یک دلیل عمده که برای عملکرد مدل در هر دو حالت میتوان اذعان کرد، بالانس نبودن دیتاست است. همانطورکه در نمودارها شکل ۱۱ مشخص است، توزیع کلاس‌ها یکسان نیست و بنابراین پتاسیل به اشتباه افتادن مدل بیشتر می‌شود.

شکل های زیر نیز دقت و هزینه را به صورت مجزا برای مدل‌ها در کنار هم نشان می‌دهند:



شکل 18. نمودارهای دقت و هزینه برای دادگان آموزشی اصلی و افزوده شده



شکل 19. نمودارهای دقت و هزینه برای دادگان اعتبارسنجی اصلی و افزوده شده

پرسش ۴. شبکه بخش‌بندی تصاویر

۴-۱. دادگان

همان‌طور که در صورت سوال ذکر شده، مجموعه داده SUIM شامل تصاویر و بخش‌بندی متناظر آن‌ها از اجسام و موجودات زیر آب است. همچنین ۱۵۲۵ داده به عنوان داده‌های تست و اعتبارسنجی و ۱۱۰ نمونه به عنوان داده‌های تست موجود هستند.

تصاویر موجودات و اجسام زیر آب به صورت RGB هستند. همچنین mask مربوط به آن‌ها به صورت mask ۸ کلاسه و با رنگ‌های مختلف و به صورت RGB است. در اینجا می‌توانیم یک نمونه از تصاویر ورودی و mask خروجی (خروجی‌ای که از مدل انتظار خواهیم داشت) را ببینیم. این نمونه از مجموعه داده‌های تست نمایش داده می‌شود.

شکل 20. اطمینان از صحت استخراج دیتا و ماسک مربوط به آن

۴-۱-۱. جداسازی داده‌های آموزش و اعتبارسنجی

حال داده‌های موجود مربوط به آموزش و اعتبارسنجی را، به داده‌های تنها برای آموزش و اعتبارسنجی تقسیم می‌کنیم. این‌کار را به صورت تصادفی انجام داده و ۱۰ درصد از این داده‌ها را برای اعتبارسنجی کنار می‌گذاریم. بهتر است تا این‌کار را به صورتی انجام دهیم تا داده‌ها به صورت متوازن بین آموزش و اعتبارسنجی تقسیم بشوند. انجام این کار، یعنی تقسیم داده‌ها به صورت متوازن فواید زیر را دارد.

- جلوگیری از عدم توازن در داده‌ها: اگر داده‌های آموزشی و ارزیابی به طور متوازن جدا نشوند، مدل ممکن است با مشکلاتی مانند نادیده گرفتن کلاس‌های کم‌تعداد یا تعمیم نادرست روبه‌رو شود. جداسازی متوازن کمک می‌کند تا مدل بتواند تمامی کلاس‌ها و نمونه‌ها را به خوبی یاد بگیرد و ارزیابی کند.

- ارزیابی دقیق‌تر عملکرد مدل: وقتی که داده‌های ارزیابی توزیع مشابهی با داده‌های آموزشی داشته باشند، عملکرد مدل در ارزیابی دقیق‌تر بازتاب‌دهنده عملکرد آن در دنیای واقعی خواهد بود. این امر به توسعه‌دهندگان مدل کمک می‌کند تا بازخورد دقیقی از عملکرد مدل دریافت کنند و در صورت نیاز، بهبودهای لازم را اعمال کنند.

- جلوگیری از سوگیری: گر جداسازی داده‌ها به صورت متوازن انجام نشود، ممکن است بایاس در مدل ایجاد شود. به عنوان مثال، اگر داده‌های آموزشی بیشتر شامل نمونه‌های یک کلاس خاص باشند، مدل ممکن است تمایل به پیش‌بینی آن کلاس داشته باشد و عملکردش در پیش‌بینی کلاس‌های دیگر ضعیف شود. جداسازی متوازن از این نوع بایاس جلوگیری می‌کند.

- تشخیص مشکلات خاص مدل: با داشتن داده‌های متوازن، می‌توان به راحتی مشکلات خاص مدل را شناسایی کرد. مثلاً اگر مدل در کلاس خاصی عملکرد ضعیفی داشته باشد، می‌توان به طور دقیق‌تری این مشکل را شناسایی و رفع کرد.

- تعمیم‌پذیری بهتر مدل: مدل‌هایی که بر روی داده‌های متوازن آموزش دیده‌اند، معمولاً تعمیم‌پذیری بهتری دارند و می‌توانند بر روی داده‌های جدید و ناشناخته بهتر عمل کنند. این به این دلیل است که مدل در طی آموزش با نمونه‌های متنوعی روبه‌رو شده و توانسته الگوهای مختلف را یاد بگیرد.

- کاهش احتمال تصادف در نتایج: جداسازی متوازن باعث می‌شود که نتایج حاصل از ارزیابی کمتر تحت تأثیر نوسانات تصادفی باشند و نتایج ارزیابی پایداری بیشتری داشته باشند.

۴-۱-۲. ساخت دیتالودر

برای ارسال داده‌ها به مدل و در واقع استفاده مدل از داده‌ها از دیتالودر استفاده می‌کنیم. در دیتالودر به این صورت عمل می‌کنیم که آدرس تصاویر مربوط به هر مجموعه مانند آموزش، تست و اعتبارسنجی را به کلاس مربوطه داده و سپس در زمان ورودی دادن آن به مدل، مدل به صورت batch-batch داده‌ها را از دیتالودر می‌گیرد که این batchها به صورت تصادفی انتخاب شده‌اند.

همچنین از زمانی که batch مشخص انتخاب شده تا به مدل برسد، تغییراتی روی داده‌ها اعمال می‌شود که این تغییرات را به ترتیب بررسی می‌کنیم.

1. در ابتدا داده‌ها از آدرس انتخاب شده، خوانده شده و به صورت ماتریس ذخیره می‌شوند (به تعداد batch)، سپس برای اینکه بتوانیم آن‌ها را به مدل ورودی دهیم، نیاز داریم تا همه تصاویر به یک سایز باشند، بنابراین همه آن‌ها را به سایز ۱۲۸×۱۲۸ می‌بریم.
2. سپس برای این‌که مدل تعمیم پذیری بهتری داشته باشد، به دلیل وجود تعداد نسبتا کمی از داده‌های آموزشی، آگمنتیشن انجام داده و داده‌ها را تا حدی نسبت به شکل اولیه آن‌ها، تغییر می‌دهیم. آگمنتیشن به صورت زیر و با احتمال‌های متناظر انجام می‌شود.

- Horizontal and vertical flip: هر کدام با احتمال ۵۰ درصد
- Random rotate 90: به احتمال ۵۰ درصد تصویر را تا ۹۰ درجه می‌چرخاند.
- Shift-Scale-Rotate: به صورت ترکیبی شیفت داده، اسیکل می‌کند و می‌چرخاند.

دلیل انتخاب هر کدام از این روش‌ها برای تقویت داده این است که، اجسام و موجودات زیر آب ممکن است به حالات مختلف درآمده و سمت و سو متفاوتی داشت باشند. این تغییرات با استفاده از این نوع از تقویت داده، به خوبی رسیدگی می‌شوند. می‌توانستیم از برخی از روش‌های تقویت داده مخصوص تصاویر زیر آب نیز استفاده کنیم، اما به نظرم ممکن بود که با انجام برخی از آن‌ها (برای مثال تغییری نسبی رنگ تصویر)، تسک بخش‌بندی به خوبی انجام نشود. بنابراین تنها به همین روش‌ها که قطعا باعث بهبود داده‌ها می‌شوند و قطعا تاثیرگذار هستند، استفاده کردیم.

3. سپس تصاویر را نرمالایز می‌کنیم. برای نرمالایز کردن، از الگورتیم min-max استفاده می‌کنیم، در واقع با تقسیم مقدار هر پیکسل بر ۲۵۵، مقادیر موجود در ماتریس بیانگر تصویر، به بازه بین ۰ تا ۱ می‌روند. نرمالایز کردن داده‌ها به بهبود عملکرد الگوریتم‌های یادگیری ماشین و شبکه‌های عصبی کمک می‌کند. همچنین تغییرات در نور و روشنایی تصویر کمتر بر روی پردازش تاثیر می‌گذارد. این روش وقتی که دقت و تطابق مقیاس‌بندی با محدوده واقعی داده‌ها مهم است، استفاده می‌شود. مثلاً در تصاویر با مقیاس‌های مختلف نور و کنتراست.

4. سپس چون بعد از نرمالایز کردن، و دقیق نبودن مقادیر موجود در ماتریس mask (در حالی که نیاز داریم برای تشخیص دقیق رنگ، این اتفاق افتاده و مقادیر به صورت strict و ۰ و ۱ باشند تا ۸ رنگ مورد نظر ساخته شود). بنابراین با استفاده از یک threshold مناسب (در اینجا ۰.۵ در نظر گرفتیم) این مقادیر را به ۰ یا ۱ می‌بریم.

5. حال نیازمند یک تغییر اساسی در دیتا (mask ها) هستیم. در حال حاضر ما maskهایی به صورت RGB، یعنی شامل ۳ فیلتر ۱۲۸×۱۲۸ هستیم که به ترتیب بیانگر رنگ‌های قرمز، سبز و آبی هستند. اما برای تعریف تابع فعال‌ساز در لایه آخر و همچنین تعریف loss function، نیازمند تغییر هستیم و نمی‌توانیم با داده به این شکل، این توابع را به آسانی تعریف کنیم. برای اینکار از این ایده استفاده می‌کنیم که برای خروجی (در واقع maskهایی که قصد تولید آن‌ها را داریم) به جای اینکه یک ماتریس ۱۲۸×۱۲۸ در ۳ خروجی دهیم که بیانگر یک ماتریس است، یک ماتریس ۱۲۸×۱۲۸ در ۸ خروجی می‌دهیم، که اگر به صورت عمقی به این ماتریس نگاه کنیم، تنها یک مقدار ۱ داشته و مابقی آن‌ها (۷ مقدار دیگر) صفر خواهند بود. با این‌کار در واقع رنگ‌ها را بسیار راحت‌تر و به قولی به صورت one-hot نمایش می‌دهیم. بنابراین کار ما برای تعریف تابع فعال‌ساز لایه آخر و تابع هزینه، بسیار آسان‌تر شده. برای تابع فعال‌ساز به این گونه عمل می‌کنیم که در لایه آخر به صورت عمقی از تابع softmax استفاده می‌کنیم. همچنین برای تابع هزینه، چون که به صورت one-hot یک ماتریس خروجی می‌دهیم (منظور از این اصطلاح صرفا این است که به صورت عمقی تنها یک مقدار یک خواهیم داشت)، بنابراین به طور مشخص باید تابع هزینه را categorical cross entropy تعریف کنیم. حال می‌توانیم با سادگی بیشتر و همچنین به صورت دقیق‌تر، یک مدل برای این داده‌ها آموزش دهیم. همچنین تبدیل این ماتریس به ماتریس RGB نیز کار سختی نخواهد بود.

۴-۲. شبکه مورد استفاده

۴-۲-۱. U-Net

مدل U-Net یک معماری شبکه عصبی است که به طور گسترده در کاربردهای پردازش تصویر مانند تقسیم‌بندی تصاویر استفاده می‌شود. ایده اصلی U-Net این است که مدل بتواند جزئیات مکان را حفظ کرده و همزمان ویژگی‌های سطح بالا را استخراج کند. این مدل شامل دو بخش اصلی encoder و decoder است که در یکی downsampling و در دیگری upsampling اتفاق می‌افتد.

یکی از ویژگی‌های اصلی U-net، استفاده از skip connectionها است. این اتصالات نقشه‌های ویژگی سطح پایین از encoder را با نقشه‌های ویژگی سطح بالا از decoder ادغام می‌کنند. اطلاعات مکانی که به دقت مکانی در سطح پیکسل کمک می‌کنند، از نقشه‌های ویژگی سطح پایین انتقال یافته و با اطلاعات متنی سطح بالا ترکیب می‌شوند. در نتیجه ویژگی‌های مکانی دقیق‌تر حفظ می‌شوند و تصویر بازسازی شده کیفیت بهتری دارد.

حال به تفسیر بخش‌های مختلف این شبکه می‌پردازیم:

1. Encoder: این قسمت به گونه‌ای وظیفه استخراج ویژگی‌ها را از تصویر ورودی دارد که این کار را با استفاده از چند لایه کانولوشنی و همچنین لایه‌های MaxPooling انجام می‌دهد که به ترتیب filter کرده و سپس scale می‌کند. همچنین مرحله به مرحله تعداد فیلترها افزایش یافته و سطح feature mapها کاهش میابند. در نتیجه یک نمایش فشرده و خلاصه از ویژگی‌های تصویر بدست می‌آید.
2. Bottleneck: شامل چند لایه (در اینجا ۲ لایه) کانولوشنی با تعداد زیادی فیلتر است که بیشترین فشرده سازی را انجام داده و یک نمایش بسیار فشرده از ویژگی‌های اصلی تصویر بدست می‌آید.
3. Decoder: وظیفه این قسمت، در واقع بازسازی تصویر mask از نمایش بسیار فشرده در bottleneck است. همچنین تعداد فیلترها در هر مرحله کاهش یافته و سطح feature mapها افزایش میابند.

بنابراین در ورودی تصویر را داده و در خروجی به کمک این ساختار و با استفاده از skip connection، mask مورد نظر را که بخش‌بندی تصویر را نشان می‌دهد، خروجی می‌دهیم.

TA U-Net .۲-۲-۴

مدل TA U-Net که به کمک لایه‌های Triplet Attention ساخته شده، یک معماری شبکه عصبیCNN است که به طور خاص برای وظایف بخش‌بندی تصویر طراحی شده است. در این مدل، هدف اصلی بهبود دقت بخش‌بندی با استفاده از مکانیسم توجه سه‌گانه (Triplet Attention) است. در ادامه به صورت کلی به ایده اصلی و عملکرد این مدل پرداخته شده است.

معماری U-Net شامل دو بخش اصلی است encoder و decoder. encoder به تدریج ویژگی‌های سطح بالاتری از تصویر ورودی استخراج می‌کند. Decoder این ویژگی‌ها را به ابعاد اولیه بازمی‌گرداند و نتیجه نهایی را به دست می‌آورد.

مکانیسم توجه سه‌گانه به منظور بهبود استخراج ویژگی‌ها و تمرکز بر نواحی مهم تصویر به کار می‌رود.

این مکانیسم شامل سه شاخه است که هر کدام توجه را در یکی از ابعاد کانال، ارتفاع و عرض به کار می‌گیرند و سپس نتایج را با هم ترکیب می‌کنند.

تفاوتی که در این مدل در مقایسه با U-Net اتفاق می‌افتد، به این صورت است که:

- Encoder: در هر بلوک، ویژگی‌ها استخراج شده و سپس به کمک لایه توجه سه‌گانه تقویت می‌شوند.
- Decoder: هر بلوک در Decoder ویژگی‌های قبلی را با ویژگی‌های متناظر در مسیر Encoder ترکیب می‌کند. این ترکیب به کمک لایه‌های CNN و لایه‌های اتصال انجام می‌شود.

مدل TA U-Net به دلیل دقت بالایی که در استخراج و بازسازی ویژگی‌ها دارد، به طور گسترده در وظایف مختلف قطعه‌بندی تصویر مانند تشخیص پزشکی، پردازش تصاویر ماهواره‌ای و سیستم‌های بینایی ماشین کاربرد دارد. مکانیسم توجه سه‌گانه باعث می‌شود تا مدل بتواند به خوبی بر نواحی مهم تصویر تمرکز کند و عملکرد بهتری نسبت به مدل‌های معمولی U-Net داشته باشد.

۴-۳. آموزش شبکه

ابرپارامترهایی که از آن‌ها در آموزش هر دو شبکه استفاده کردیم به این صورت است:

- Optimizer = Adam
- Learning rate = 0.0005
- Loss function = categorical cross entropy
- Epoch = 50

البته قابل ذکر است که loss function ترکیبی ذکر شده داخل مقاله که به صورت زیر است نیز پیاده‌سازی شده، اما به دلیل عملکرد نه چندان خوب، از همان categorical cross entropy استفاده کردیم.

$$L_{(Cross\text{-}entropy)} = -\frac{1}{p}\sum_{i=1}^{p} log f_i(y^*_i)$$

$$f_i(c) = \frac{e_i^F(c)}{\sum_c^f \in c e_i^F(c^i)}, i \in [1,p], c \in C$$

$$\widetilde{y}_i = argmax F_i(c)$$

$$J_c(y^*, \widetilde{y}) = \frac{|(y^* = c) \cap (\widetilde{y} = c)|}{|((y^* = c) \cup (\widetilde{y} = c)|}$$

$$L_{Lovasz\text{-}Softmax} = \Delta J_c(y^*, \widetilde{y}) = 1 - J_c(y^*, \widetilde{y})$$

$$a + b = 1$$

$$L = aL_{Lovasz\text{-}Softmax} + bL_{(Cross\text{-}entropy)}$$

شکل 21. تابع هزینه ترکیبی

همچنین یک مورد قابل ذکر دیگر نیز که قبلا هم بررسی کردیم، این است که تابع فعال‌ساز لایه آخر هر دو این مدل‌ها softmaxای بوده که به صورت عمقی عمل می‌کند، نه به صورت تک مقداری.

حال هر مدل را و همچنین summary آن را بررسی می‌کنیم.

U-Net .۱-۳-۴

ساختار پیاده‌سازی شده U-Net به صورت زیر است.

Layer (type)	Output Shape	Param #	Connected to
input_layer_15 (InputLayer)	(None, 128, 128, 3)	0	-
conv2d_285 (Conv2D)	(None, 128, 128, 64)	1,792	input_layer_15[0…
conv2d_286 (Conv2D)	(None, 128, 128, 64)	36,928	conv2d_285[0][0]
max_pooling2d_60 (MaxPooling2D)	(None, 64, 64, 64)	0	conv2d_286[0][0]
conv2d_287 (Conv2D)	(None, 64, 64, 128)	73,856	max_pooling2d_60…
conv2d_288 (Conv2D)	(None, 64, 64, 128)	147,584	conv2d_287[0][0]
max_pooling2d_61 (MaxPooling2D)	(None, 32, 32, 128)	0	conv2d_288[0][0]
conv2d_289 (Conv2D)	(None, 32, 32, 256)	295,168	max_pooling2d_61…
conv2d_290 (Conv2D)	(None, 32, 32, 256)	590,080	conv2d_289[0][0]
max_pooling2d_62 (MaxPooling2D)	(None, 16, 16, 256)	0	conv2d_290[0][0]
conv2d_291 (Conv2D)	(None, 16, 16, 512)	1,180,160	max_pooling2d_62…
conv2d_292 (Conv2D)	(None, 16, 16, 512)	2,359,808	conv2d_291[0][0]
max_pooling2d_63 (MaxPooling2D)	(None, 8, 8, 512)	0	conv2d_292[0][0]
conv2d_293 (Conv2D)	(None, 8, 8, 1024)	4,719,616	max_pooling2d_63…
conv2d_294 (Conv2D)	(None, 8, 8, 1024)	9,438,208	conv2d_293[0][0]
conv2d_transpose_60 (Conv2DTranspose)	(None, 16, 16, 512)	2,097,664	conv2d_294[0][0]
concatenate_60 (Concatenate)	(None, 16, 16, 1024)	0	conv2d_transpose… conv2d_292[0][0]

conv2d_295 (Conv2D)	(None, 16, 16, 512)	4,719,104	concatenate_60[0…
conv2d_296 (Conv2D)	(None, 16, 16, 512)	2,359,808	conv2d_295[0][0]
conv2d_transpose_61 (Conv2DTranspose)	(None, 32, 32, 256)	524,544	conv2d_296[0][0]
concatenate_61 (Concatenate)	(None, 32, 32, 512)	0	conv2d_transpose… conv2d_290[0][0]
conv2d_297 (Conv2D)	(None, 32, 32, 256)	1,179,904	concatenate_61[0…
conv2d_298 (Conv2D)	(None, 32, 32, 256)	590,080	conv2d_297[0][0]
conv2d_transpose_62 (Conv2DTranspose)	(None, 64, 64, 128)	131,200	conv2d_298[0][0]
concatenate_62 (Concatenate)	(None, 64, 64, 256)	0	conv2d_transpose… conv2d_288[0][0]
conv2d_299 (Conv2D)	(None, 64, 64, 128)	295,040	concatenate_62[0…
conv2d_300 (Conv2D)	(None, 64, 64, 128)	147,584	conv2d_299[0][0]
conv2d_transpose_63 (Conv2DTranspose)	(None, 128, 128, 64)	32,832	conv2d_300[0][0]
concatenate_63 (Concatenate)	(None, 128, 128, 128)	0	conv2d_transpose… conv2d_286[0][0]
conv2d_301 (Conv2D)	(None, 128, 128, 64)	73,792	concatenate_63[0…
conv2d_302 (Conv2D)	(None, 128, 128, 64)	36,928	conv2d_301[0][0]
conv2d_303 (Conv2D)	(None, 128, 128, 8)	520	conv2d_302[0][0]
softmax_15 (Softmax)	(None, 128, 128, 8)	0	conv2d_303[0][0]

Total params: 31,032,200 (118.38 MB)
Trainable params: 31,032,200 (118.38 MB)
Non-trainable params: 0 (0.00 B)

این مدل دقیقا همانند TA U-Net موجود در مقاله است که بدون Triplet Attention پیاده سازی شده است.

TA U-Net .۲-۳-۴

ساختار این شبکه نیز دقیقا مانند تصویر موجود در مقاله و به صورت زیر است.

Layer (type)	Output Shape	Param #	Connected to
input_layer_33 (InputLayer)	(None, 128, 128, 3)	0	-
conv2d_645 (Conv2D)	(None, 128, 128, 64)	1,792	input_layer_33[0…
batch_normalizatio… (BatchNormalizatio…	(None, 128, 128, 64)	256	conv2d_645[0][0]
activation_443 (Activation)	(None, 128, 128, 64)	0	batch_normalizat…
conv2d_646 (Conv2D)	(None, 128, 128, 64)	36,928	activation_443[0…
batch_normalizatio… (BatchNormalizatio…	(None, 128, 128, 64)	256	conv2d_646[0][0]
activation_444 (Activation)	(None, 128, 128, 64)	0	batch_normalizat…
max_pooling2d_114 (MaxPooling2D)	(None, 64, 64, 64)	0	activation_444[0…
conv2d_647 (Conv2D)	(None, 64, 64, 128)	73,856	max_pooling2d_11…
batch_normalizatio… (BatchNormalizatio…	(None, 64, 64, 128)	512	conv2d_647[0][0]
activation_445 (Activation)	(None, 64, 64, 128)	0	batch_normalizat…
conv2d_648 (Conv2D)	(None, 64, 64, 128)	147,584	activation_445[0…
batch_normalizatio… (BatchNormalizatio…	(None, 64, 64, 128)	512	conv2d_648[0][0]
activation_446 (Activation)	(None, 64, 64, 128)	0	batch_normalizat…
triplet_attention_… (TripletAttention)	(None, 64, 64, 128)	8,832	activation_446[0…
max_pooling2d_115 (MaxPooling2D)	(None, 32, 32, 128)	0	triplet_attentio…
conv2d_652 (Conv2D)	(None, 32, 32, 256)	295,168	max_pooling2d_11…

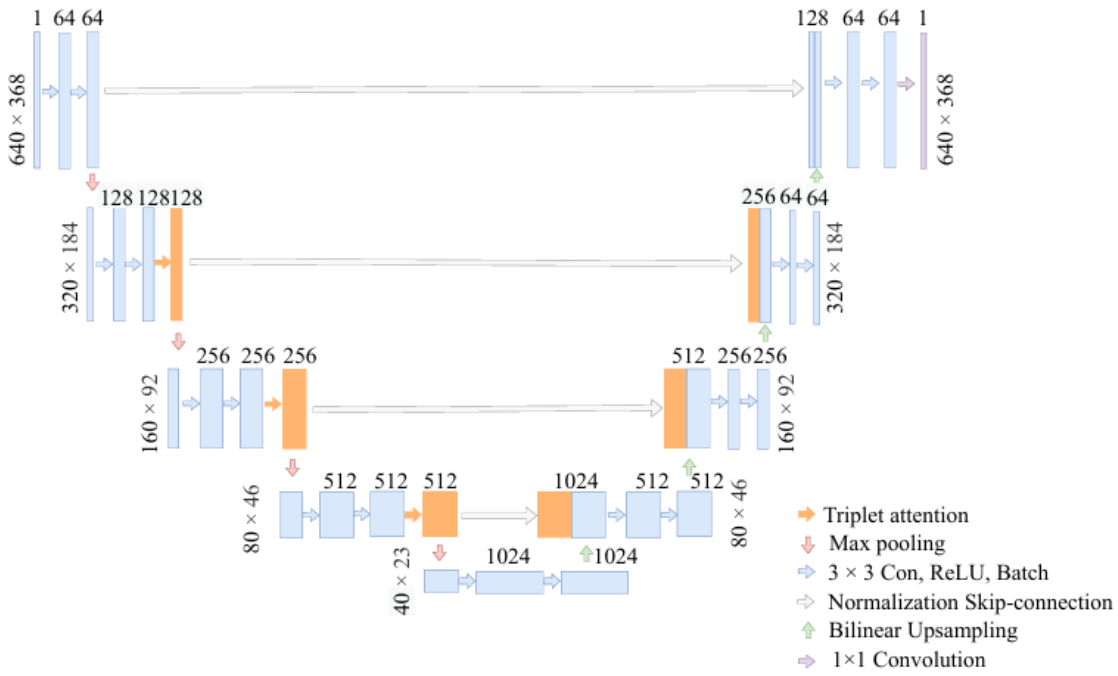
batch_normalizatio… (BatchNormalizatio…	(None, 32, 32, 256)	1,024	conv2d_652[0][0]
activation_450 (Activation)	(None, 32, 32, 256)	0	batch_normalizat…
conv2d_653 (Conv2D)	(None, 32, 32, 256)	590,080	activation_450[0…
batch_normalizatio… (BatchNormalizatio…	(None, 32, 32, 256)	1,024	conv2d_653[0][0]
activation_451 (Activation)	(None, 32, 32, 256)	0	batch_normalizat…
triplet_attention_… (TripletAttention)	(None, 32, 32, 256)	17,664	activation_451[0…
max_pooling2d_116 (MaxPooling2D)	(None, 16, 16, 256)	0	triplet_attentio…
conv2d_657 (Conv2D)	(None, 16, 16, 512)	1,180,160	max_pooling2d_11…
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 512)	2,048	conv2d_657[0][0]
activation_455 (Activation)	(None, 16, 16, 512)	0	batch_normalizat…
conv2d_658 (Conv2D)	(None, 16, 16, 512)	2,359,808	activation_455[0…
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 512)	2,048	conv2d_658[0][0]
activation_456 (Activation)	(None, 16, 16, 512)	0	batch_normalizat…
triplet_attention_… (TripletAttention)	(None, 16, 16, 512)	35,328	activation_456[0…
max_pooling2d_117 (MaxPooling2D)	(None, 8, 8, 512)	0	triplet_attentio…
conv2d_662 (Conv2D)	(None, 8, 8, 1024)	4,719,616	max_pooling2d_11…
batch_normalizatio… (BatchNormalizatio…	(None, 8, 8, 1024)	4,096	conv2d_662[0][0]
activation_460 (Activation)	(None, 8, 8, 1024)	0	batch_normalizat…
conv2d_663 (Conv2D)	(None, 8, 8, 1024)	9,438,208	activation_460[0…
batch_normalizatio… (BatchNormalizatio…	(None, 8, 8, 1024)	4,096	conv2d_663[0][0]

activation_461 (Activation)	(None, 8, 8, 1024)	0	batch_normalizat…
conv2d_transpose_1… (Conv2DTranspose)	(None, 16, 16, 512)	2,097,664	activation_461[0…
concatenate_104 (Concatenate)	(None, 16, 16, 1024)	0	conv2d_transpose… triplet_attentio…
conv2d_664 (Conv2D)	(None, 16, 16, 512)	4,719,104	concatenate_104[…
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 512)	2,048	conv2d_664[0][0]
activation_462 (Activation)	(None, 16, 16, 512)	0	batch_normalizat…
conv2d_665 (Conv2D)	(None, 16, 16, 512)	2,359,808	activation_462[0…
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 512)	2,048	conv2d_665[0][0]
activation_463 (Activation)	(None, 16, 16, 512)	0	batch_normalizat…
conv2d_transpose_1… (Conv2DTranspose)	(None, 32, 32, 256)	524,544	activation_463[0…
concatenate_105 (Concatenate)	(None, 32, 32, 512)	0	conv2d_transpose… triplet_attentio…
conv2d_666 (Conv2D)	(None, 32, 32, 256)	1,179,904	concatenate_105[…
batch_normalizatio… (BatchNormalizatio…	(None, 32, 32, 256)	1,024	conv2d_666[0][0]
activation_464 (Activation)	(None, 32, 32, 256)	0	batch_normalizat…
conv2d_667 (Conv2D)	(None, 32, 32, 256)	590,080	activation_464[0…
batch_normalizatio… (BatchNormalizatio…	(None, 32, 32, 256)	1,024	conv2d_667[0][0]
activation_465 (Activation)	(None, 32, 32, 256)	0	batch_normalizat…
conv2d_transpose_1… (Conv2DTranspose)	(None, 64, 64, 128)	131,200	activation_465[0…
concatenate_106 (Concatenate)	(None, 64, 64, 256)	0	conv2d_transpose… triplet_attentio…
conv2d_668 (Conv2D)	(None, 64, 64,	295,040	concatenate_106[…

	128)		
batch_normalizatio… (BatchNormalizatio…	(None, 64, 64, 128)	512	conv2d_668[0][0]
activation_466 (Activation)	(None, 64, 64, 128)	0	batch_normalizat…
conv2d_669 (Conv2D)	(None, 64, 64, 128)	147,584	activation_466[0…
batch_normalizatio… (BatchNormalizatio…	(None, 64, 64, 128)	512	conv2d_669[0][0]
activation_467 (Activation)	(None, 64, 64, 128)	0	batch_normalizat…
conv2d_transpose_1… (Conv2DTranspose)	(None, 128, 128, 64)	32,832	activation_467[0…
concatenate_107 (Concatenate)	(None, 128, 128, 128)	0	conv2d_transpose… activation_444[0…
conv2d_670 (Conv2D)	(None, 128, 128, 64)	73,792	concatenate_107[…
batch_normalizatio… (BatchNormalizatio…	(None, 128, 128, 64)	256	conv2d_670[0][0]
activation_468 (Activation)	(None, 128, 128, 64)	0	batch_normalizat…
conv2d_671 (Conv2D)	(None, 128, 128, 64)	36,928	activation_468[0…
batch_normalizatio… (BatchNormalizatio…	(None, 128, 128, 64)	256	conv2d_671[0][0]
activation_469 (Activation)	(None, 128, 128, 64)	0	batch_normalizat…
conv2d_672 (Conv2D)	(None, 128, 128, 8)	520	activation_469[0…
softmax_21 (Softmax)	(None, 128, 128, 8)	0	conv2d_672[0][0]

Total params: 31,117,576 (118.70 MB)
Trainable params: 31,100,424 (118.64 MB)
Non-trainable params: 17,152 (67.00 KB)

بنابراین مدل‌های را با استفاده از دیتالودر تعریف شده برای داده‌های آموزش و اعتبارسنجی آموزش می‌دهیم. نتایج آن‌ها در بخش تحلیل نتایج موجود است.



شکل 22. **TA U-Net** موجود در مقاله

۴-۴. ارزیابی و تحلیل نتایج

۴-۴-۱. MeanIoU

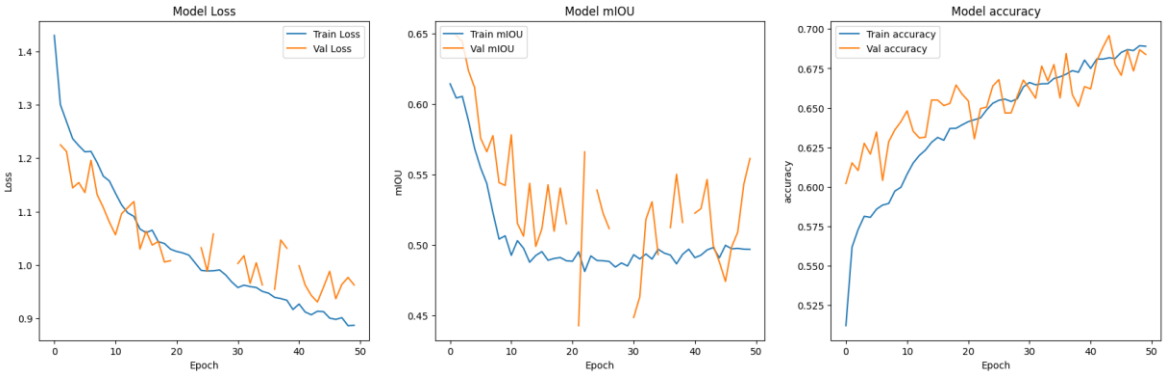
معیار Mean Intersection over Union (MeanIoU) یکی از معیارهای ارزیابی مدل‌های بخش‌بندی در Computer Vision است. این معیار برای ارزیابی دقت بخش‌بندی تصویر استفاده می‌شود و به مدل کمک می‌کند تا بداند چقدر خوب توانسته است پیکسل‌های مختلف را به درستی دسته‌بندی کند.

همانطور که از اسم آن پیداست، مقدار آن با تقسیم مساحت ناحیه مشترک بر مساحت ناحیه اجتماع بدست می‌آید. در کاربردهای واقعی، معمولاً با چندین کلاس مختلف در تفکیک‌بندی تصویر سروکار داریم. برای محاسبه MeanIoU ، IoU برای هر کلاس جداگانه محاسبه شده و میانگین آنها گرفته می‌شود.

MeanIoU به عنوان یک معیار ارزیابی جامع به مدل کمک می‌کند تا عملکرد خود را در تمام کلاس‌های موجود در تصویر به طور متوازن بهبود بخشد. این معیار به ویژه در کاربردهای پزشکی، ماشین‌های خودران و سیستم‌های تشخیص چهره بسیار مهم است، زیرا دقت تفکیک‌بندی در این حوزه‌ها بسیاد مهم می‌باشد.

۴-۴-۲. نمودار آموزش

در ابتدا نمودار مربوط به مدل U-Net را بررسی کنیم.

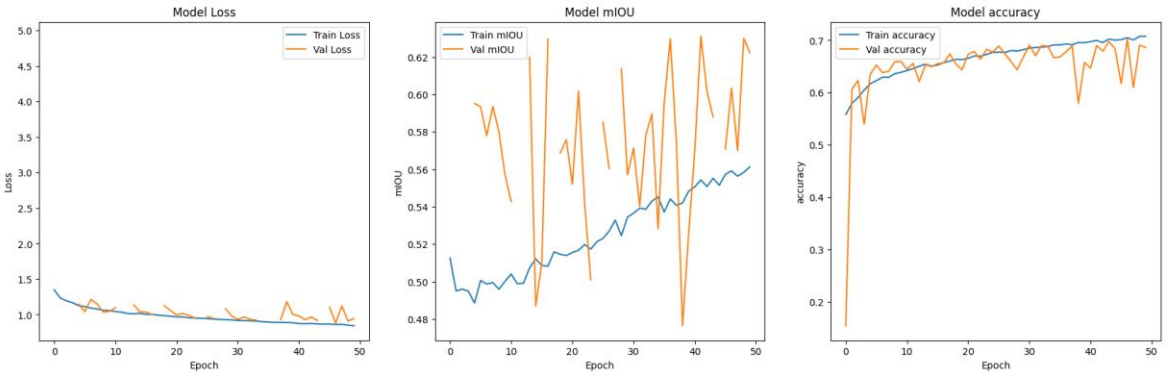


شکل 23. آموزش **U-Net**

همانگونه که مشخص است، مدل همواره در حال بهتر کردن دقت خود و همچنین کم کردن مقدار تابع هزینه است. اما از طرفی مقدار mIoU همان ابتدا کم شده و در همان حدود ۵۰ درصد باقی‌مانده. همچنین mIoU مربوط به داده‌های اعتبارسنجی در اکثر مواقع بهتر و بیش‌تر از داده‌های آموزش است.

این اتفاق که mIoU کاهش یافته و دیگه افزایش پیدا نکرده، در حالی که هم هزینه در حال کم‌تر شدن و هم دقت در حال بالا رفتن است، بسیار عجیب بوده و نتونستم دلیل این اتفاق رو متوجه بشوم.

حال به بررسی آموزش شبکه TA U-Net می‌پردازیم.



شکل 24. آموزش شبکه **TA U-Net**

همانظور که قابل مشاهده است، در اینجا دوباره دقت و هزینه در حال بهتر شدن بودند، با این تفاوت نسبت به U-Net که در ابتدا از مقدار بیشتری شروع شدند. تفاوت اساسی (در واقع مهم‌ترین تفاوت) که در آموزش این شبکه نسبت به U-Net شاهد هستیم، این است که در اینجا mIoU از حدود ۵۰ درصد شروع شده و در حال افزایش است که هدف از آموزش شبکه نیز همین اتفاق می‌باشد.

در بعضی از جاها مقدار loss و یا mIoU مقدار nan گرفته و متوجه نشدم که چه اتفاقی باعث این میشه.

۴-۴-۳. mIoU بر روی داده‌های تست

ابتدا برای U-Net بررسی می‌کنیم.

```
accuracy: 0.6800 - loss: 0.9216 - mean_iou: 0.4614
```

که این میزان در همان حدودی است که برای داده‌های اعتبار سنجی از مدل دیدیم.

حال این مقادیر را برای TA U-Net بررسی می‌کنیم.

```
accuracy: 0.6751 - loss: 0.9550 - mean_iou: 0.5717
```

که دوباره این مقادیر تقریبا برابر مقادیر متناظر برای داده‌های اعتبارسنجی است.

۴-۴-۴. کدام شبکه برای بخش‌بندی

همانطور که مشاهده می‌کنیم و کاملا مشخص است، TA U-Net عملکرد خیلی بهتری نسبت به U-Net عادی در بخش Mean IoU داشته در حالی که دقت و تابع هزینه هر دو تقریبا دارای مقادیر یکسانی است.

بنابراین با توجه به نتایج و با توجه به تکنیک‌های مورد استفاده از TA U-Net، این شبکه در اکثر مواقع باید عملکرد بهتری نسبت به U-Net عادی داشته باشد. بنابراین انتخاب من برای بخش‌بندی، این مدل خواهد بود. این انتخاب هم دلیل در اهمیت زیاد معیار mIoU است چرا که در واقع که معیار است که بیانگر شبیه بودن خروجی تولید شده به خروجی واقعی است.