به نام خدا

ملیکا احمدی رنجبر 97521036

1) Representation Learning با توجه به درس مطرح شده در کلاس، برای ایجاد بازنمایی‌هایی بهتر از Input Data ما است، بازنمایی‌هایی که ما را برای تشخیص Feature و Classification کمک می‌کند. در واقع این روش به جای Feature Engineering که در گذشته انجام می‌شد است، و برای بدست آوردن Featureها با استفاده از Representation جدید در هر لایه برای Input است، حال چه عکس باشد، چه متن و .. . Representation Learning می‌تواند هم Supervised و هم Unsupervised باشد. Supervised زمانی است که Data ورودی ما همراه با Label و Class موجود برای Training است و این مرحله با استفاده از این Labelها انجام می‌شود و Featureها را نیز متناسب با آن بدست می‌آورد. در واقع یک جفت ورودی داریم و باید Input را به Label با توجه به Featureهایی که بدست می آوریم Map کنیم.

در قسمت Unsupervised ورودی ما بدون Label وارد Model ما می‌شود و مرحله آموزش و Feature Extraction باید بدون Label باشد و System باید تماما خودش Pattern و Feature برای تشخیص در Data را آموزش بیند.

از طرف دیگر مبحث Self-Supervised Learning را داریم که حد واسط Supervised و Unsupervised است، به این ترتیب که ما با استفاده از Unsupervised Technique در ابتدا یک Model بزرگ و یک Dataset بزرگ (بدون Label) را آموزش می‌دهیم، و سپس از Featureهای آن برای آموزش موارد دیگری که Supervised هستند استفاده می‌کنیم (در واقع General Features را آموزش می‌بیند و بعدا آن را برای حل موارد دیگر به کار می‌گیرد). برای مثال پر کردن و تشخیص بخش حذف شده از عکس، یا مسئله جورچین .. . در این دسته، نکته مهم آن است، که Model سعی خواهد داشت که شباهات و اختلافات را بیاید و با استفاده از آن مسائل دیگر را حل کند (برای مثال همان تصاویری که شباهت بین اساتید را بررسی کردیم، یا همچنین پر کردن جای خالی در جمله و پیدا کردن Word2Vec).

به طور کلی، در نهایت از این Modelها برای Fine Tuning کردن یک Down Streaming Task استفاده کرد.

2) با مطالعه مقاله:

- Utility در این سوال، برابر است با تعداد Labelهایی که برای دست‌یابی به دقت با Self-Supervised لازم است برای روش Baseline استفاده شود. و در واقع مقدار Label بیشتری که نیاز است تا Model به دقت Self-Supervised که Fine Tuned شده، برسد. برای مثال زمانی که که Label بیشتری نیاز نیز و Already مقادیر Accuracy برابر دارند Utility = 0 است. و زمانی Utility برابر Infinite می‌شود که تحت هیچ شرایط Model دیگر به Accuracy ما در Fine Tuned Model دست نیابد.
- در ابتدا می‌بینیم که Taskها به طور کلی یا توجه به معنا و مفهوم دارند مثل Reorganizing Object Categories، یا هندسه مثل Surface Normal Estimation. در اینجا

Object Classification, Object Pose Estimation, Semantic Segmentation, Depth Estimation

را داریم. و به طور کلی تفاوت Classification/Segmentation و Pose/Depth را می‌خواهیم بررسی کنیم برای تفاوت Taskهای معنایی و هندسی، و همچنین برای Global/Dense Features.

Object Classification: تصاویر به گونه ای تولید می شوند که فقط شامل یک شی هستند و توزیع یکنواختی در سراسر وجود دارد.

Object Pose Estimation: تصاویر دوباره به گونه ای تولید می شوند که فقط شامل یک شی هستند و توزیع یکنواختی در سراسر وجود دارد. و به جای در نظر گرفتن چرخش کامل شی، آن را به 5 دسته تقسیم کرده و Train می کنیم. (Upward, Forward, Downward, Left, Right)

Semantic Segmentation: چند شی خواهیم داشت و Resolution نیز در اینجا بالا خواهد بود، Per Pixel است (Loss).

Depth Estimation: همانند Semantic Segmentation است که دارای چند شی برای آموزش است و همچنین Dense.

- در اینجا نیز، 4 Algorithm مختلف برای Pertaining داریم:

Variational Autoencoder: برای Map کردن تصویر ورودی به یک فضای نمونه با Dimension کم.
Rotation: تشخیص چرخش در تصاویر 0و 90و 180و 270 درجه.
Contrastive Multiview Coding: تقسیم تصویر به چند کانال مختلف، از دو شبکه نصف شده عبور کرده و Embeddingهای خروجی با هم مقایسه می‌شوند.
Augmented Multiscale Deep InfoMax: همچون روش قبل است. فقط به جای مقایسه کردن Channelها AMDIM بازنمایی هایی را از دو نسخه Augment شده یک تصویر و همچنین بازنمایی‌های تولید شده در Intermediate Layers مقایسه می کند.

3) در این سوال، قسمت‌های مشخص شده را با کمک دیدن برخی نمونه‌ها تکمیل نمودم. در بخش اول، لازم بود تا هر یک از اطلاعات لازم را به بردار صحیح آن تبدیل کنیم. برای این این بخش، که بتوانیم Story, Query, Answer را به صورت جداگانه مشخص کنیم، ابتدا برای هر سه یک List در نظر گرفته، تا در نهایت همان را بازگردانیم. سپس برای هر Triplet Tuple که در Data Variable موجود است، For می‌زنیم و برای هر یک، بردار Numpy در نظر می‌گیریم. سپس، با توجه به شماره کلمه‌ای که به ترتیب در هر کدام استفاده شده، بردارهای جدید را ساخته، و به List نهایی که برای خروجی است، Append می‌کنیم. برای Answer نیز، صرفا کلمه مشخص شده را برابر 1 قرار می‌دهیم.
در مرحله بعد لازم بود تا مقادیر Loss, Accuracy را با توجه به Modelای که Fit شده است، Plot کنیم، که این کار را نیز بارها انجام داده‌بودیم (History را به عنوان ورودی می‌گیریم).

در نهایت نیز نوبت به پیاده‌سازی Model رسم شده رسید. در اینجا نیز تنها نکته به نسبت سخت نحوه Code زدن آن بود، با توجه به داشتن چند Sequential. در واقع کاری که در اینجا انجام شد، در نظر گرفتن مقادیر ورودی Query و Story بود، که هر کدام را به یک لایه Embedding داده (در واقع لایه Sequential مطرح شده در تصویر دارای Embedding است، که Output Dimension آن نیز معلوم شده، Input Dimension را نیز که از قبل با توجه به تعداد کل کلمات داریم). یکی دیگر ار لایه‌های مهم و اصلی LSTM است، که در واقع یک RNN است با در نظر گرفتن کلمات قدیمی‌تر. خروجی این لایه نیز تنها یک مقدار انتهایی است، و آن را به لایه Fully Connected می‌دهیم، تا در نهایت با استفاده از Activation Function = Softmax برای اینکه از بین 22 کلمه انتخاب شود، جواب را بدست آوریم. با توجه به تعداد Epoch کم برای این مسئله، Accuracy بدست آمده مقدار زیادی نسیت و برای بهبود عملکرد می‌توان تعداد Iteration را افزایش یافت تا Model درک بهتری از Input Sequence و به طور کلی سوال داشته باشد. از طرفی برای بهبود Model و لایه‌های مورد استفاده در آن می‌توان با توجه به مباحث گفته شده در کلاس گفت، استفاده از Conv و LSTM نیز می‌تواند نتیجه را بهبود بخشد، با توجه به اینکه در ابتدا Featureهای Local برای Sequence ما بدست می‌آورد و سپس آن را به LSTM می‌دهیم. اگرچه این مسئله به دلیل Vocabulary Size کم با همین Model اما در Epochهای بیشتر تیز می‌تواند نتیجه بهتری داشته‌باشد. عملکرد در داده‌های Test نیز بهتر بوده است، و با توجه به دقت پایین هر دو حالت، احتمالا Model دچار Under Fitting شده‌است. در انتها نیز یک مثال دلخواه را امتحان می‌کنیم:

```
Result
John moved to the hallway . Mary travelled to the office . Where is Mary ? | Prediction: office
```

4) منابع:

- https://keras.io/api/layers/core_layers/embedding/
- https://keras.io/api/layers/reshaping_layers/permute/
- https://www.techtarget.com/searchenterpriseai/definition/unsupervised-learning
- https://en.wikipedia.org/wiki/Unsupervised_learning
- https://en.wikipedia.org/wiki/Self-supervised_learning#:~:text=Self%2Dsupervised%20learning%20(SSL),on%20an%20artificial%20neural%20network.&text=Second%2C%20the%20actual%20task%20is%20performed%20with%20supervised%20or%20unsupervised%20learning.
- https://en.wikipedia.org/wiki/Feature_learning
- https://towardsdatascience.com/supervised-semi-supervised-unsupervised-and-self-supervised-learning-7fa79aa9247c