گزارش سوال دو

امیرعباس افضلی – 400100662

توضیح الگوریتم FF به صورت خلاصه :

در این الگوریتم سعی شده که از انجام EBP جلوگیری شود و نحوه ی اپتیمایز کردن پارامتر ها لایه به لایه اتفاق می افتد . یعنی با انجام تعدادیepoch یک لایه ترین می شود و سپس همین کار، برای لایه لایه ی بعید انجام می شود و ... .

همچنین هم در مود supervised وهم unsupervised ، داده های مثبت و منفی تشکیل می شود و با استفاده از این داده ها و تشکیل goodness برای آنها ، یک loss تعریف می شود و پارامتر های هر لایه نسبت به این لاس اپتیمایز می شوند.

به گفته ی خود مقاله :

“ Each layer has its own objective function which is simply to have high goodness for positive data and low goodness for negative data.”

Goodness definition:

$$p(positive) = \sigma\left(\sum_j y_j^2 - \theta\right)$$

Loss function:

$$loss = mean(log(1 + e^{[(threshold - positivedata),(negativedata - threshold)]}))$$

مشاهده می شود که تابع هزینه ی بالا ، تابعی صعودی برحسب goodness دیتای مثبت و تابعی نزولی بر حسب goodness دیتای منفی است.

و ایده آل ما نیز این است که goodness ای که از خروجی لایه بدست می آید ، برای دیتای مثبت زیاد و برای دیتای منفی کم باشد . بنابراین این تعریف تابع هزینه با این هدف همخوانی دارد و پارامتر ها به نحوی اپتیمایز می شوند که این هدف را برآورده کنند.

Mean هم بدلیل این است که ترین به صورت mini-batch انجام می شود.

در ترین شبکه در مود supervised از 128 batch-size استفاده شده است.

نتایج :

train accuracy: 93.945

test accuracy: 93.829

دقت شود که چون در این شبکه ، loss برای هر لایه تعریف می شود ، نمی توانیم یک مقدار loss کلی برای داده های ترین و تست (مانند سایر شبکه های forward-backward) تعریف کنیم .

اگر معیار خطا را به صورت (loss = 100 – accuracy) تعریف کنیم ، داریم :

train loss: 6.055

test loss: 6.171

توضیح الگوریتم :

ابتدا dataloader مناسب با batch-size = 128 برای داده های ترین و تست تعریف می کنیم . برای تولید positive & negative دیتا ، از توابع prepare_positive_data و prepare_negative_data که پیاده سازی کردیم ، استفاده می کنیم (همانطور که در مقاله گفته شده که 10 پیکسل اول را به بردار one-hot لیبل اختصاص می دهیم که البته در داده های منفی ، این لیبل ، لیبل اصلی داده نمی باشد)

سپس کلاس layer را تعریف می کنیم و همانطورر که گفته شد ، ایپاک های مختلف روی هر لایه انجام می شود و پارامتر های آن لایه اپتیمایز می شوند. (از ADAM استفاده شده است)

سپس کلاس شبکه ی اصلی را تعریف می کنیم که لایه های کلاس ، object هایی از کلاس layer می باشند.

برای predict کردن هم ، لیبل های مختلف روی یک دیتا می گزاریم و مجموع goodness لایه های مختلف که به ازای این دیتا تولید کرده اند را حساب می کنیم . لیبلی که این مجموع در آن بزرگ تر باشد را به عنوان لیبل predict شده دیتا انتخاب می کنیم.

و در نهایت با کمک دیتا لودر شبکه (در معماری شبکه از دو لایه استفاده کردیم) را ترین می کنیم و سپس بر روی دیتای ترین و تست پردیکشن را انجام می دهیم.

Unsupervised

همانطور که در مقاله آمده است ، در حالت unsupervised ، از شبکه به عنوان یک embedder استفاده می شود و در نهایت با استفاده از representation ای که شبکه از دیتا آموخته است و خروجی داده است، با استفاده از یک لایه dense و softmax ، طبقه بندی را انجام می دهیم.

فایده این کار در آن است که در دیتا ست های پیچده تر از mnist مانند SIFAR و غیره ، انجام طبقه بندی صرفا با یک linear clf ساده تقریبا نتایج مطلوبی نخواهد داشت و با استفاده از این شبکه FF در مود unsupervised ، یک embedding مناسبی از دیتا ایجاد می کنیم و می توانیم طبقه بندی را با یک linear clf ساده انجام دهیم.

به بیان ساده تر شبکه ی ما در حالت آنسوپروایز ، دیتا ی کلاس های مختلف را به نحوی در فضایی n بعدی map می کند که discriminate کردن دیتا به کلاس های مختلف ساده تر انجام شود.

بیان خود مقاله :

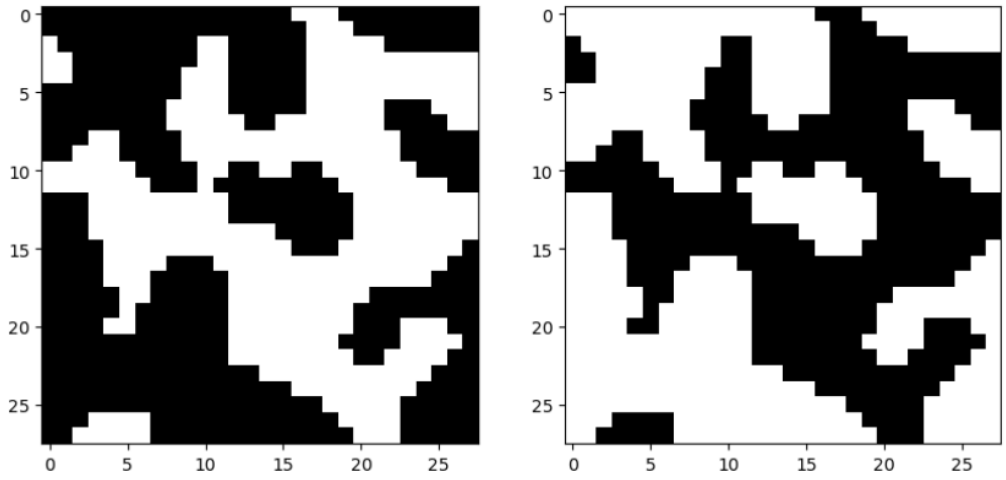
Learning hidden representations without using any label information is quite sensible for large models that eventually need to be able to perform a wide variety of tasks. The unsupervised learning extracts a smorgasbord of features and the individual tasks can use whichever features are helpful. But if we are only interested in one task and we want to use a small model that does not have the capacity to model the full distribution of the input data, it makes more sense to use supervised learning. One way to achieve this with FF is to include the label in the *input*. The positive data consists of an image with the correct label and the negative data consists of an image with the incorrect label. Since the only difference between positive and negative data is the label, FF will ignore all features of the image that do not correlate with the label.

در این بخش ساختار شبکه تقریبا ثابت است و صرفا تابع predict شبکه تغییر می کند (زیرا نیاز داریم خروجی لایه های مختلف را به ازای یک دیتا ی ورودی cash کنیم و در نهایت این خروجی لایه های مختلف را concat می کنیم و به عنوان ورودی به linear clf می دهیم).

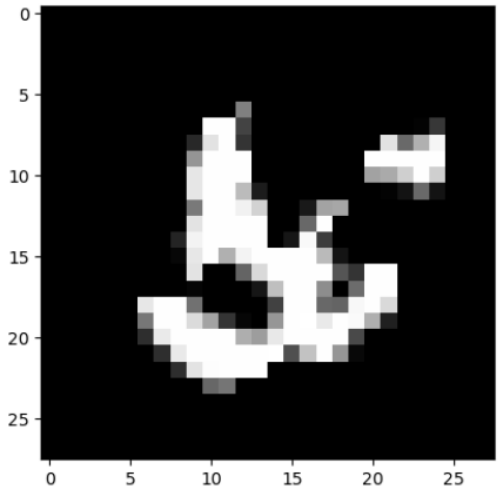
همیچنین دیگر 10 پیکسل اول داده های مثبت را تغییر نمی دهیم و دیتا ی منفی را هم همانطور که در مقاله آمده است ایجاد می کنیم .

To force FF to focus on the longer range correlations in images that characterize shapes, we need to create negative data that has very different long range correlations but very similar short range correlations. This can be done by creating a mask containing fairly large regions of ones and zeros. We then create hybrid images for the negative data by adding together one digit image times the mask and a different digit image times the reverse of the mask as shown in figure 1. Masks like this can be created by starting with a random bit image and then repeatedly blurring the image with a filter of the form $[1/4, 1/2, 1/4]$ in both the horizontal and vertical directions. After repeated blurring, the image is then thresholded at 0.5.

یک نمونه از ماسک های تولید شده :



یک نمونه از hybrid data ی تولید شده :



نتایج طبقه بندی linear classifier :

train accuracy: 91.364

test accuracy: 90.913

train loss: 8.636

test loss: 9.087

linear classifier نهایی ، در 100 ایپاک به صورت supervised ترین شده است.