|  |  |  |
| --- | --- | --- |
|  | به نام خدا |  |
| **دانشگاه تهران**  **دانشکده‌ی مهندسی برق و کامپیوتر**  **شبکه عصبی**  **گزارش پروژه ۳** | | |

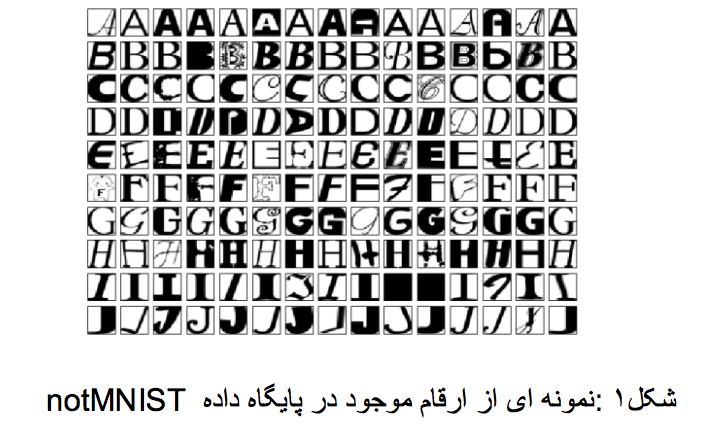
|  |  |  |
| --- | --- | --- |
| ملیکه احقاقی | سید علی طباطبایی آل طه | نام و نام خانوادگی |
| ۸۱۰۱۹۴۲۵۴ | ۸۱۰۱۹۴۴۶۲ | شماره‌ی دانشجویی |
| ۲۷/۲/۹۷ | | تاریخ ارسال گزارش |

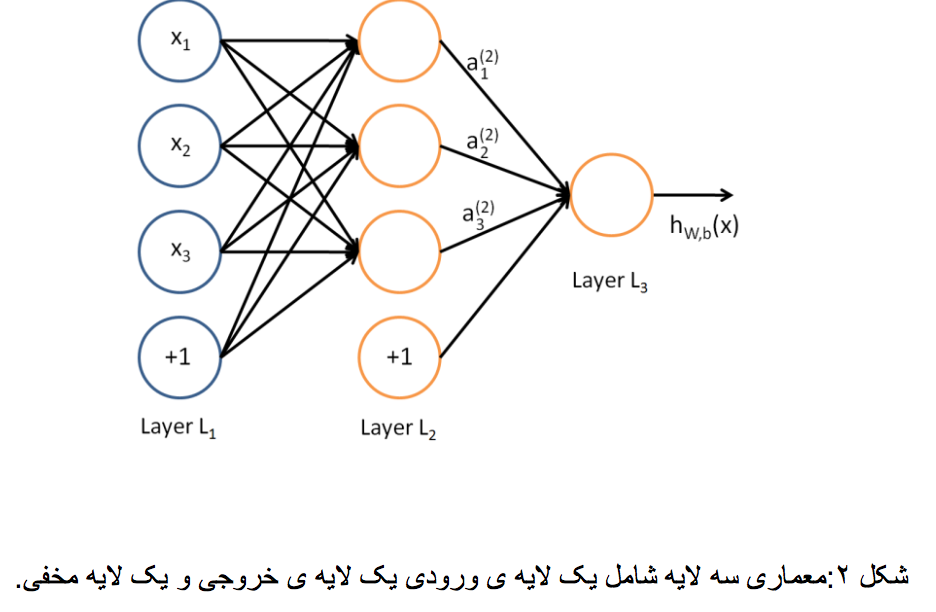
­



* 1. **چکیده**

یک شبکه عصبی مصنوعی ایده ای برای پردازش اطلاعات است که از سیستم عصبی زیستی الهام گرفته و مانند مغز به پردازش اطلاعات می پردازد. این سیستم از شمار زیادی عناصر پردازش به هم پیوسته به نام نورون تشکیل شده که برای حل یک مساله باهم هماهنگ عمل می کنند. شبکه های عصبی مصنوعی با پردازش داده های تجربی، دانش یا قانون نهفته در ورای داده ها آنها را به ساختار شبکه منتقل می کنند که به این عمل یادگیری می گویند.اصولا توانایی یادگیری مهم ترین ویژگی یک سیستم هوشمند است. سیستمی که قابلیت یادگیری داشته باشد، منعطف تر است وساده تر برنامه ریزی می شود، بنابراین بهتر می تواند در مورد مسایل و معادلات جدید پاسخگو باشد. در این ساختار با ایجاد شبکه ای بین این گره ها و اعمال یک الگوریتم آموزشی به آن، شبکه را آموزش می دهند. در این حافظه یا شبکه عصبی گره ها (نورونها) دارای دو حالت فعال (روشن یا ۱) و غیرفعال (خاموش یا ۰) اند و هر یال (سیناپس یا ارتباط بین گره ها) دارای یک وزن میباشد. یالهای با وزن مثبت، موجب تحریک یا فعال کردن گره غیر فعال بعدی می شوند و یالهای با وزن منفی، گره متصل بعدی را غیرفعال یا مهار (در صورتی که فعال بوده باشد) می کنند.در واقع تنظیم وزنهای ورودی هر نورون عصبی باعث یادگیری کل شبکه می شود. پس از اینکه مجموعه ای از داده ها به منظور آموزش به شبکه داده می شود با هر بار دریافت خروجی و مقایسه با جواب مورد انتظار اگر به تشخیص صحیح رسیده بود مسیر هایی که به این تشخیص منجر شده است از طریق شبکه تقویت می شود (از طریق نرمالیزه کردن وزن یالها).شبکه های عصبی مصنوعی می توانند دارای لایه های متعددی باشند و یا یک لایه باشند. از کاربرد های شبکه عصبی عبارت اند از حل مسایل شناسایی الگو و پردازش تصویر و متن , پردازش زبان های طبیعی , مسایل دسته بندی و ... است . در این پروژه پس از پیاده سازی شبکه ی عصبی حروف Aتا j را دسته بندی می نماییم .هدف از پیاده سازی این پروژه آموزش شبکه ی عصبی در جهت تشخیص حروف براساس تصویری از یک حرف است .





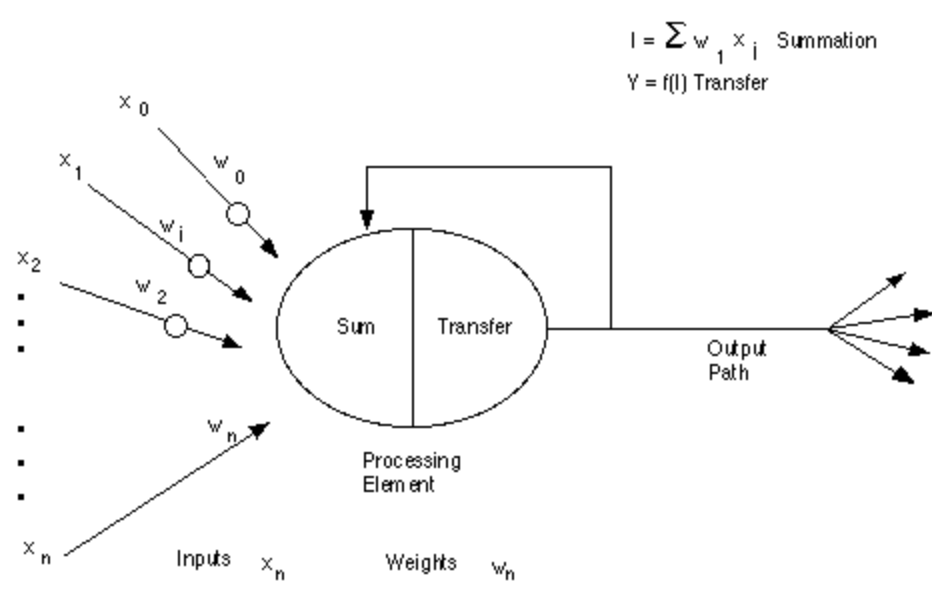
* 1. **ارائه‌ی روش**

پیاده سازی این شبکه ی عصبی از چند فاز تشکیل می شود.ورودی های این شبکه شامل test **و** trainingاز مجموعه داده هایnotMINISTاستفاده می کند که با تعدادی از تصاویر مانند شکل ۱ شبکه را آموزش می دهیم و با کمک تصاویری مشابه آن را تست می نماییم. این ورودی ها تصاویر ۲۸\*۲۸ پیکسل هستند که در نتیجه لایه ی ورودی از ۷۸۴ نورون متناظر این پیکسل ها تشکیل می شود.در ابتدا توسط کاربر انتخاب هایی صورت می گیرد که اولا روش بهینه سازی GD ویا SGD باشد. هم چنین کاربر می تواند تابع فعال ساز را از بین تابع خطی و sigmoid انتخاب نماید.نهایتا روش regularization نیز از بین L2Norm و dropout توسط کاربر مشخص خواهد شد.بسته به انتخاب هایی که کاربر دارد آموزش و تست شبکه عصبی شروع می شود.با دریافت option های کاربر و تشکیل training\_set و test\_set شبکه تشکیل خواهد شد.این شبکه از سه لایه ی ورودی و خروجی همچنین یک لایه ی درونی تشکیل شده است.

لایه ی ورودی با دریافت training\_set و test\_set initialize می شود. در این جا چک می شود که اگر روش بهینه سازی stochastic باشد ابتدا training\_set را shuffle می کند. همچنین لایه ی میانی و خروجی نیز به ترتیب از ۳۰(عدد تجربی)و ۱۰ نورون تشکیل می شود. که این ۱۰ نورون متناظر با هریک از حروف خواهد بود. در ۱۰۰۰ iteration مراحل train و test تکرار خواهد شد.

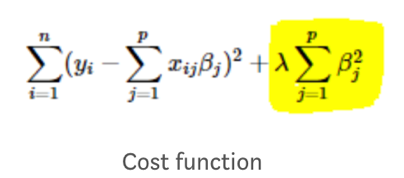
**Train**

در این مرحله به تعداد عناصر training set آموزش شبکه صورت می گیرد.در هرمرحله تصویر مربوطه به همراه نام حرف دریافت می شود.در صورتی که انتخاب کاربر روش dropout باشد برای اعداد ورودی یک احتمال تصادفی بین ۰ و۱ انتخاب می شود که ۰ بودن آن به منزله ی حذف نورون متناظر است.سپس محاسبات لایه میانی انجام می شود که بسته به اینکه linear ویا sigmoid باشد خروجی نورون های میانی محاسبه می شود.

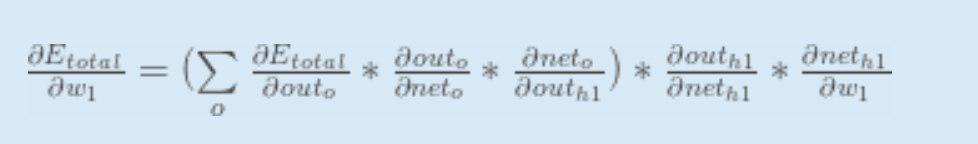


همین مراحل برای لایه ی خروجی انجام می شود.سپس loss\_function را محاسبه می نماییم که از فرمول زیر محاسبه می شود:

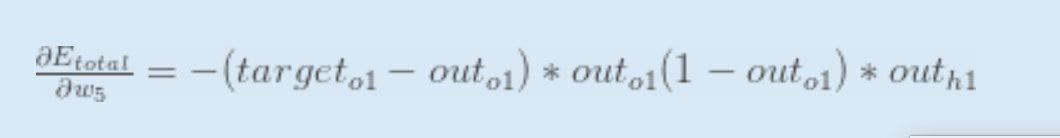
در صورتی که روش regularization انتخابی L2Norm باشد این مقدار به صورت زیر محاسبه می شود:



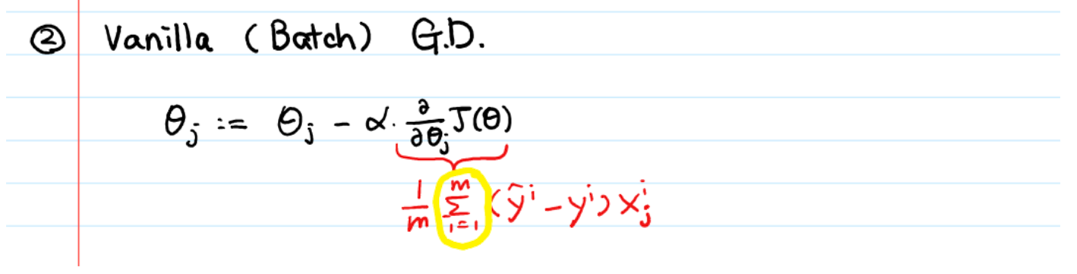
سپس backpropagation روی لایه ی خروجی و بعد لایه ی میانی انجام می شود. برای انجام این مرحله درصورتی که تابع SGD باشد در لایه ی ورودی به میانی از فرمول زیر به منظور محاسبه ی delta استفاده می شود:

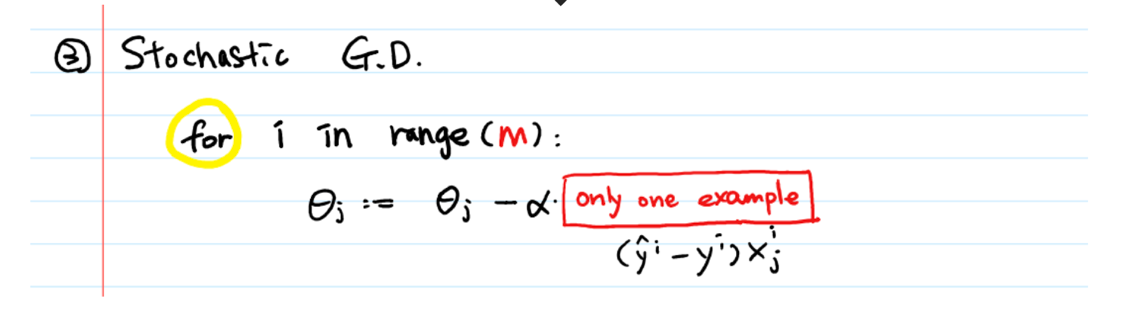


برای محاسبه ی delta بین لایه ی میانی و خروجی نیز از فرمول زیر بهره برده ایم:

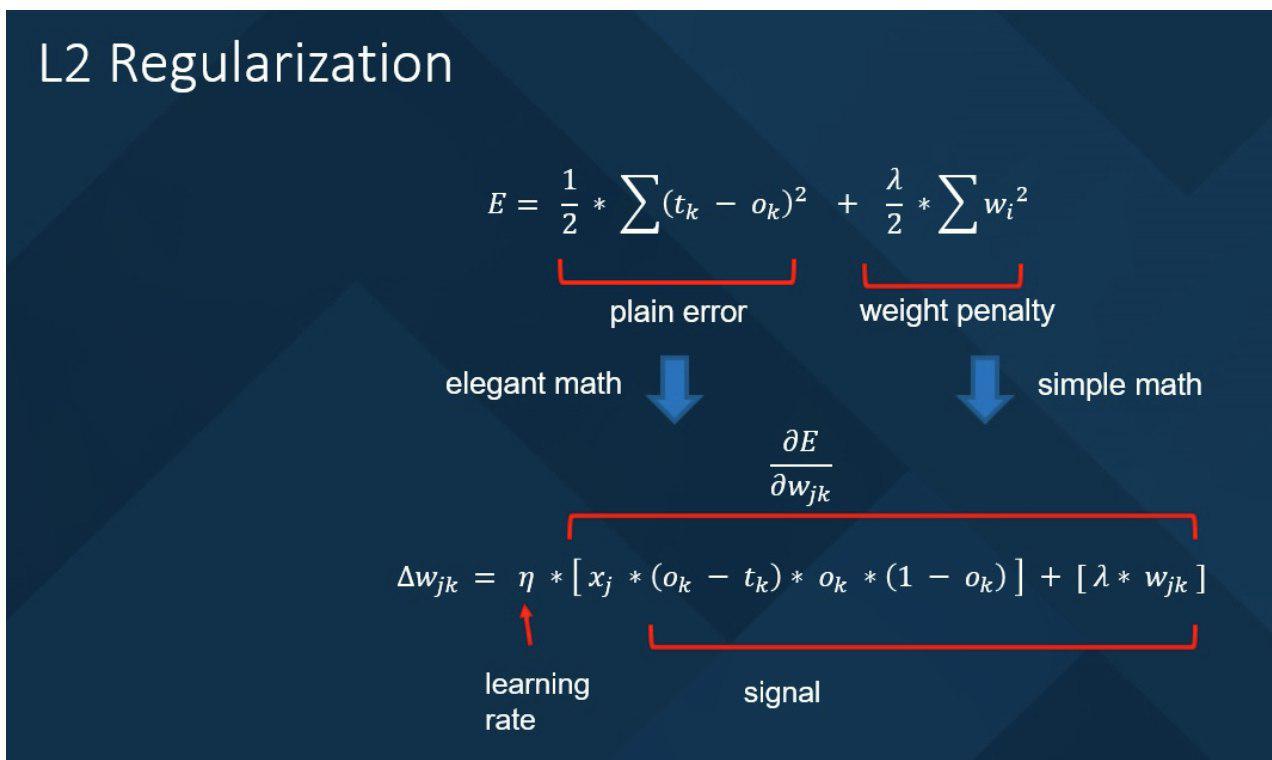


تفاوت روش SGD و GD در شکل های زیر قابل ملاحظه است:





با بهره گیری از این فرمول ها و همچنین اعمال l2Norm درصورت انتخاب شدن آن توسط کاربر مراحل backpropagation به پایان خواهد رسید.



**Test**

بعد از پایان مرحله ی قبل وارد فاز test خواهیم شد. در این مرحله در طی یک feed forward در طول لایه ها خروجی شبکه را در ازای هر ورودی تست به دست می آوریم.با توجه به ماتریس ده تایی خروجی حدس شبکه در مورد حرف را هم ارز با خروجی با نزدیکترین عدد به ۱ در نظر می گیریم.در صورتی که predict در هر iteration برابر با خروجی مطلوب بود به تعداد خروجی های درست یکی می افزاییم.در نهایت برای به دست آوردن دقت شبکه از درصد predict های درست نسبت به تعداد کل تست ها استفاده می کنیم.

در نهایت بعد از اتمام train و test با محاسبه ی loss function به طور میانگین می توان نمودار loss function را با کمک LossAccPlotter() به نمایش گذاشت.

****



* 1. **ارائه‌ی نتایج**
  2. **تحلیل نتایج**
  3. **جمع‌بندی و نتیجه‌گیری**
  4. **پاسخ سوالات تءوری**

۱) در مرحله ی feed-forward از لایه ی input شروع و در طول لایه ها به انجام محاسبات روی آن می پردازیم و خروجی واقعی مدل را نهایتا در لایه ی خروجی به دست می آوریم. از آن جا که جریان محاسبات از لایه ی ورودی به سمت لایه ی خروجی انجام می شود به این مرحله feed-forward گفته می شود.

در مرحله ی feed-backward در واقع از loss function استفاده می کنیم و با انجام مشتق گیری های لازم از نقطه ی پایان به سمت نقطه شروع حرکت می کنیم و update های لازم را در وزن ها و بایاس ها ایجاد می کنیم.

**Backpropagation Steps**:

Input -> Forward calls -> Loss function -> derivative -> backpropagation

۲)

* **Supervised:**

در یادگیری با ناظر ممکن است روابط داخلی میان داده های در حال پردازش از پیش مشخص نباشد اما ما می دانیم که هدف ما مدل کردن چه خروجی خواهد بود. آن چه از پیش نمی دانیم این هست که خروجی مطلوب را **چگونه** به دست آوریم. بنابراین شما می توانید مجموعه ای از داده های موجود را برای **"training"** یک مدل برای پیش بینی ها یا پاسخ های خاص استفاده کنید. آموزش مدل معمولا از بخشی از داده ها برای **یادگیری** استفاده می کند و بخشی از داده ها برای **اعتبار سنجی و اندازه گیری** دقیق مدل است.

* **Unsupervised:**

در الگوریتم های بدون ناظر، هنوز نمی دانید که خروجی مدل چه خواهد بود. شما احتمالا حدس می زنید که برخی روابط بین داده های شما وجود دارد، اما داده ها برای حدس زدن بسیار پیچیده است. بنابراین در این موارد شما داده های خود را **normalize** می کنید که قابل مقایسه باشند و اجازه می دهید که مدل به کار خود ادامه دهد و برخی از این روابط را به دست آورد. یکی از ویژگی های خاص این مدل این است که در حالی که می تواند راه های مختلفی برای دسته بندی یا مرتب سازی داده های شما پیشنهاد کند، این برعهده شماست که اجازه دهید که جست و جو را ادامه داده و نتایج مفید بیشتری به دست آورد. ادامه ی این فعالیت می تواند به عنوان تقویت اطلاعات راجع به روابط داخلی تلقی شود.

* **Reinforcement:**

این الگوریتم مساله ها و تکنیک هایی را که به منظور ارتقاء آن تلاش می کند، مدل سازی می کند. به منظور انجام این کار، RL نیاز به توانایی senseدارد و به صورت خودکار تصمیم گیری در مورد یک action انجام می دهد و سپس نتیجه را با یک تعریف ازreward مقایسه می کند. RL تلاش می کند تا چگونگی به حداکثر رساندن اینreward ها را بفهمد، اما این کار را به خودی خود (با دستورالعمل مستقیم) انجام نمی دهد. Agent با توجه به حالت environment و شرایط action ای را انتخاب می کند که به بیشترین پاداش را دریافت کند. این action ها environment و state عامل را تغییر می دهند. هم چنین با دادن پاداش به agent وقفه در آن ها ایجاد خواهد شد. با اجرای این حلقه و تکرار شدن آن agent رفتار خود را بهبود می بخشد.

.

۳) **overfitting** :

این مشکل زمانی پیش می آید که آنالیزی از یک مجموعه داده ی خاص به دست آید به گونه ای که دقیقا و با اختلاف خیلی کمی با آن دسته ی خاص تطابق دارد. همان طور که در شکل زیر مشاهده می شود خط سیاه یک مدل **regularized** و خط سبز یک مدل **overfitted** از داده ها ست که بیش ازبه آن دسته خاص از داده وابسته است و احتمالا در ازای مواجه شدن با داده ی ورودی جدید با خطای بیش تری مواجه می شود.

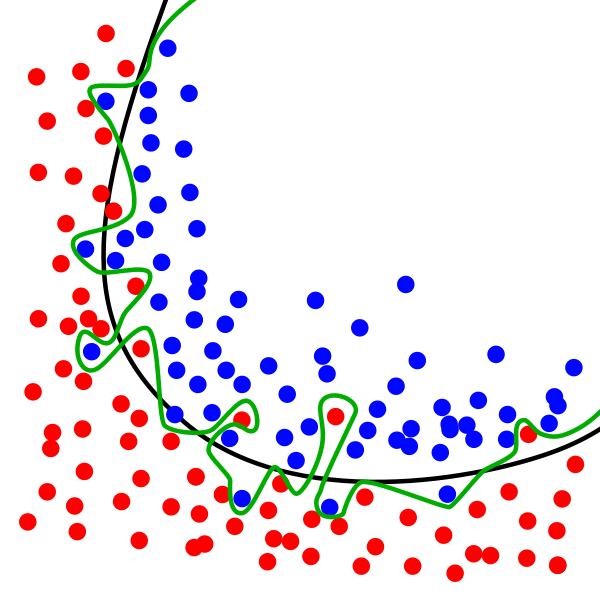


Figure2-overfitting

برای مشکل overfitting راه حل های زیادی وجود دارد که در ادامه به یک سری از آن ها اشاره شده است:

* Cross Validation

در روش استاندارد k-fold cross-validation داده ها را به k دسته افراز می نماییم. K-1 دسته ی اول را در یک حلقه ی k-1 تایی train می کنیم و از داده های دسته ی k به عنوان test استفاده می کنیم.

* Train more data

این روش همیشه پاسخگو نیست اما training با داده ی بیشتر کمک میکند که سیگنال بهتری detect شود. این تکنیک با اضافه شدن داده های noisy راه حل مناسبی نخواهد بود.

* Remove features

یک سری از الگوریتم ها یک انتخاب built-in از feature ها دارند. به این صورت که به صورت manual یک دسته از input feature های غیر مرتبط را به منظور بالابردن generalizability حذف می نماییم.

* Early stopping

در iteration های مختلف از جایی به بعد توانایی generalizion مدل کم می شود و خروجی شروع به overfit می کند. در این الگوریتم قبل از رسیدن به آن نقطه جلوی iteration های بعدی گرفته خواهد شد.

* Regularization
* Ensembling

.

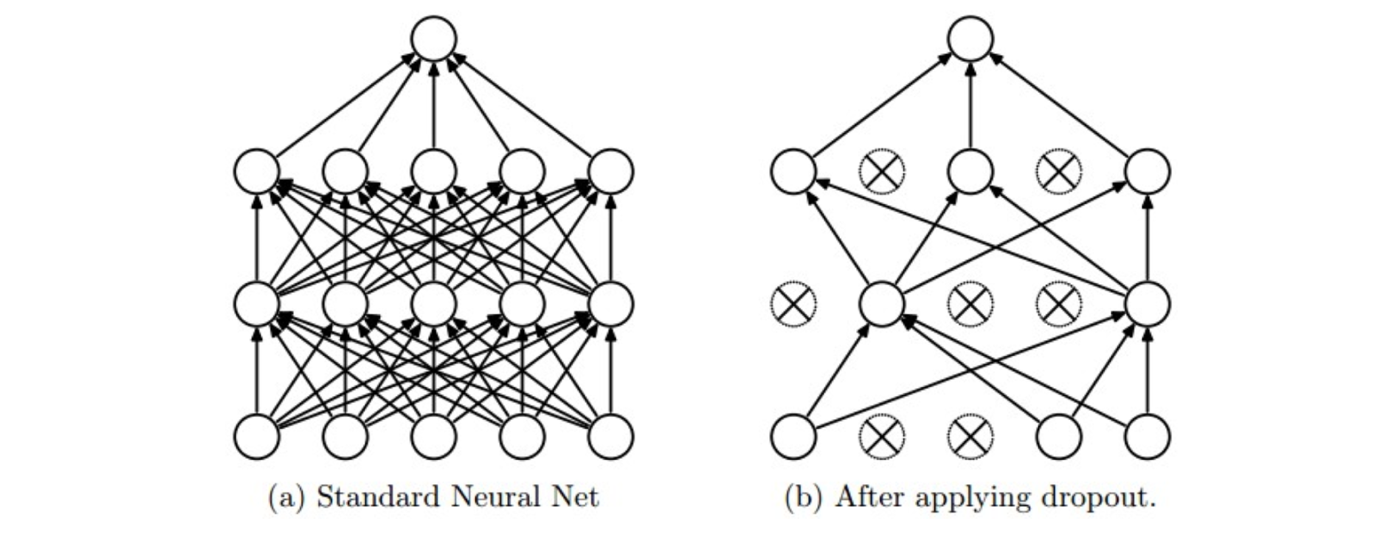
.

.

۴)

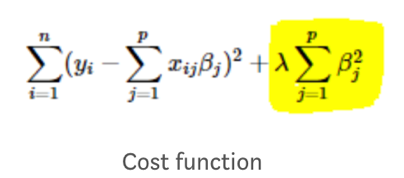
* Drop out

در این تکنیک با احتمال p=1-q تعدادی از گره های شبکه را از آن خارج می کنیم به این صورت که تمام اتصالات آن به لایه ی قبل و بعد نادیده گرفته شود.در این روش به طور موقت تعدادی از نورون ها را غیر فعال می نماییم. این روش در فاز training به منظور کاهش overfit به کار گرفته می شود. با dropout شبکه می تواند تمرکز بیش تری روی feature های دیگر داشته باشد.



* L2Norm

در این روش مربع اندازه ی ضرایب به عنوان penalty term به تابع اضافه میشود. بخش highlight شده المان regularization را در روش L2 نمایش می دهد. در صورتی که لامبدا برابر صفر باشد به روش قبل بازمی گردیم و در صورتی که خیلی بزرگ باشد وزن زیادی را اضافه می کند که موجب underfitting خواهد شد. پس انتخاب لامبدا اهمیت پیدا می کند.

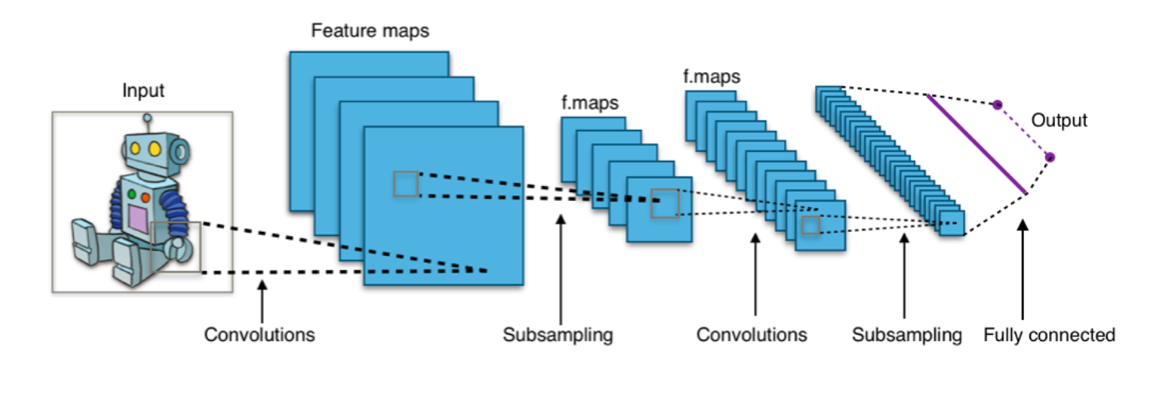


فایده ی اصلی dropout در این است که اجازه می دهد که یک شبکه تعداد زیادی از زیرشبکه ها را به صورتی کم هزینه و آسان شامل دو بخش traing و test را مدل می کند. نتایج آزمایشات نشان می دهد که dropout نه تنها کارایی بهتری در شبکه های بزرگ دارد بلکه از روش L2 دارای robustness بیش تری خواهد شد. در مقابل L2 دقت پیش بینی بالاتری در شبکه های کوچک تر دارد چرا که میانگین learning model می تواند کارایی کلی را وقتی تعداد submodel ها زیاد است بالا ببرد و هر یک از این submodel ها بایستی از دیگری متفاوت باشد.

۵) در بخش نتایج خروجی های loss function نمایش داده شده است.

۶)

* **Convolutional**



در این گونه شبکه های عصبی از یک یا بیشتر لایه ی convolutional تشکیل شده است که هر یک از یک convolution operation روی ورودی که نتایج را به لایه ی بعدی دهد استفاده می کنند و این عملیات اجازه می دهد که شبکه ای عمیق تر با تعداد کم تری از پارامترها ایجاد شود. این گونه شبکه ها خروجی های بسیار خوبی در اپلیکیشن های تصویری و صوتی دارند.

* **Recurrent**

در این گونه شبکه ها اتصالات بین نورون ها به گونه ای است که یک حلقه ی جهت دار ایجاد می کنند. این به این معناست که خروجی نه تنها به ورودی فعلی بلکه به مراحل قبلی state فعلی نورون وابسته است. این حافظه به حل مساءل NLP همچون connected handwriting recognition پیوسته و یا speech recognition کمک می کند.

مقایسه:

**CNN**

۱-CNN تعداد ثابت ورودی می گیرد و تعداد ثابتی خروجی می دهد.

۲-CNN یک نوع از feed-forward که تنوعی از perceptron های چندلایه دارند که به منظور استفاده در حجم کمی از پیش پردازش طراحی شده اند.

۳-CNN ها از الگو های اتصال بین نورون ها استفاده می کنند که الهام گرفته از ساختار visual cortex در حیوانات است که نورون های آن به این منظور ترتیب داده شده اند که به نواحی overlapping که کنار یک دیگر می تواند یک visual field را تشکیل دهند پاسخ می دهند.

۴-CNN ها در پردازش تصاویر و ویدیو ها ایده آل هستند.  
**RNN**

۱-RNN طول ورودی و خروجی اش دلخواه است.

۲-RNN برخلاف feedforward از حافظه ی داخلی خود برای پردازش دنباله ی دلخواهی از ورودی ها استفاده می کند.

۳-این گونه شبکه ها از یک سری اطلاعات در طول زمان بهره می گیرند.(time-series)

۴-RNN ها در تحلیل متن و کلام ایده آل هستند.