|  |  |  |
| --- | --- | --- |
| **پرسش‌های ۱ و ۲** | **نام دستیار طراح** | پویا صادقی |
| **رایانامه** | pouyasadeghi2012@gmail.com |
| **پرسش‌های ۳ و ۴** | **نام دستیار طراح** | حدیثه مصباح |
| **رایانامه** | hadisehmesbah@gmail.com |
|  | **مهلت ارسال پاسخ** | **۱۴۰3.۰1.20** |

|  |  |  |
| --- | --- | --- |
|  | **به نام خدا**  **دانشگاه تهران**  **دانشکده‌ مهندسی برق و کامپیوتر** |  |
| **درس شبکه‌های عصبی و یادگیری عمیق**  **تمرین اول** | | |

**فهرست**

[**قوانین** 1](#_Toc163165387)

[**پرسش 1**. **شبکه محاسبه مکمل 2** 1](#_Toc163165388)

[۱-۱. ساختار شبکه 1](#_Toc163165389)

[۱-2. پیاده‌سازی در پایتون 2](#_Toc163165390)

[**پرسش ۲** **– حملات خصمانه در شبکه‌های عصبی** 4](#_Toc163165391)

[۱-۲. **آشنایی با مجموعه دادگان** 4](#_Toc163165392)

[2-2. ایجاد و آموزش مدل 5](#_Toc163165393)

[3-2. پیاده‌سازی حمله FGSM 6](#_Toc163165394)

[4-2. پیاده‌سازی حمله PGD 7](#_Toc163165395)

[**پرسش ۳** **– عنوان پرسش سوم به فارسی** 9](#_Toc163165396)

[۱-۳. **عنوان بخش اول** 9](#_Toc163165397)

[**پرسش ۴** **– عنوان پرسش چهارم به فارسی** 10](#_Toc163165398)

[۱-۴. **عنوان بخش اول** 10](#_Toc163165399)

**شکل‌ها**

شکل 1. ساختار شبکه با نورون‌های Mcculloch Pitts 1

شکل 2. اعمال تمام حالات ممکن ورودی و خروجی متناظر آن‌ها 3

شکل 3. نمونه‌ی هر کلاس از مجموعه دادگان MNIST 4

شکل 4. هیستوگرام داده‌های آموزش و آزمون 4

شکل 5. مشخصات مدل پیاده شده 5

شکل 6. نمودار خطا بر اساس پارامتر epsilon در حمله FGSM 6

شکل 7. نمونه داده‌ها و لیبل واقعی و خروجی مدل در حمله FGSM 6

شکل 8. نمونه‌های آزمایش 5 و حمله PGD 8

**جدول‌ها**

[جدول 1. وزن‌ها در شبکه 2](#_Toc163165411)

[جدول 2. نتایج حمله PGD 7](#_Toc163165412)

# **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 فرآيند تصحيح از اهميت ويژه­اي برخوردار است**؛ بنابراین، لطفا تمامی نکات و فرض­هایی را كه در پیاده­سازی­ها و محاسبات خود در نظر مي­گيريد 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:

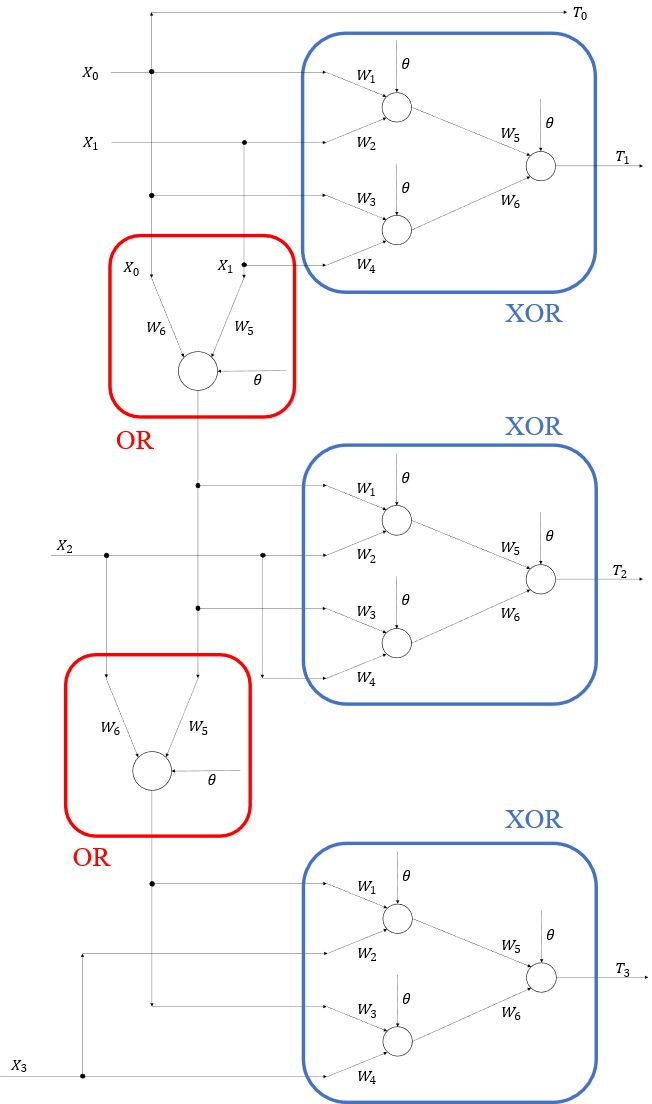
HW[Number] \_[Lastname]\_[StudentNumber]\_[Lastname]\_[StudentNumber].zip

(مثال: HW1\_Ahmadi\_810199101\_Bagheri\_810199102.zip)

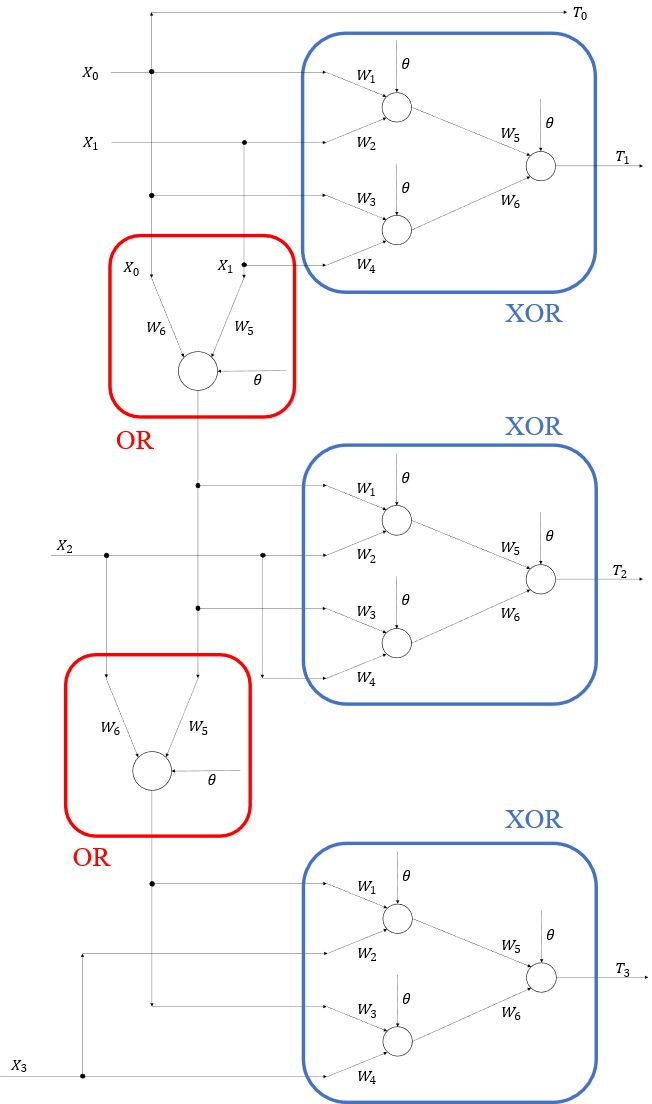
* برای گروه‌های دو نفره، بارگذاری تمرین از جانب یکی از اعضا کافی است ولی پیشنهاد می‌شود هر دو نفر بارگذاری نمایند.

# **پرسش 1**. **شبکه محاسبه مکمل 2**

## ۱-۱. ساختار شبکه



شکل 1. ساختار شبکه با نورون‌های Mcculloch Pitts

همانطور که در 

شکل 1 دیده می‌شود در این شبکه از 11 نورون Mcculloch Pitts استفاده و ساختار شبکه مطابق طراحی سطح گیت انجام شده است. (در شکل معادل عصبی هر گیت با رنگ‌های آبی و قرمز به ترتیب برای گیت XOR و OR مشخص شده است.)

جدول 1. وزن‌ها در شبکه

|  |  |
| --- | --- |
|  | **وزن** |
| **W1** | 1+ |
| **W2** | 1- |
| **W3** | 1- |
| **W4** | 1+ |
| **W5** | 1+ |
| **W6** | 1+ |

مقدار آستانه برای تمام نورون‌ها 1+ و تابع فعال ساز مطابق زیر در نظر گرفته شده است:

## ۱-2. پیاده‌سازی در پایتون

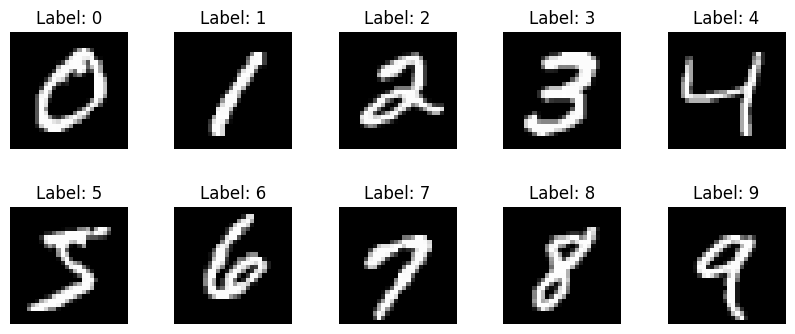
برای پیاده‌سازی از کتابخانه خارجی استفاده نشده است. یک کلاس به نام Neuron ایجاد شده که اطلاعات وزن‌ها و آستانه را در constructor خود می‌گیرد. هر نورون با استفاده از تابع Connect می‌تواند به نورون‌های دیگر وصل شود یا با تابع AsFix ورودی‌های ثابت داشته باشد.

در ابتدای ساخت شبکه هر بیت در ورودی باینری به یک نورون ثابت (AsFix) تبدیل شده و سپس شبکه‌ای متشکل از نورون‌های به هم متصل با تابع Connect متصل می‌شود. برای آزمودن کد تمام 16 حالت ممکن به شبکه داده شده که خروجی‌ها در شکل مشاهده می‌شود.

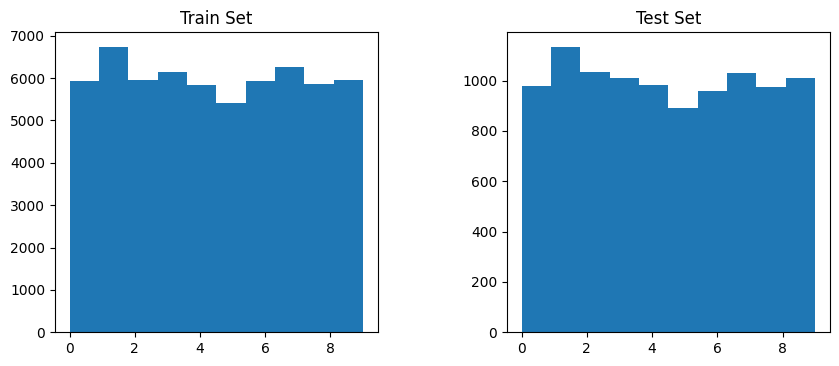


شکل 2. اعمال تمام حالات ممکن ورودی و خروجی متناظر آن‌ها

# **پرسش ۲** **– حملات خصمانه در شبکه‌های عصبی**



شکل 3. نمونه‌ی هر کلاس از مجموعه دادگان MNIST



شکل 4. هیستوگرام داده‌های آموزش و آزمون

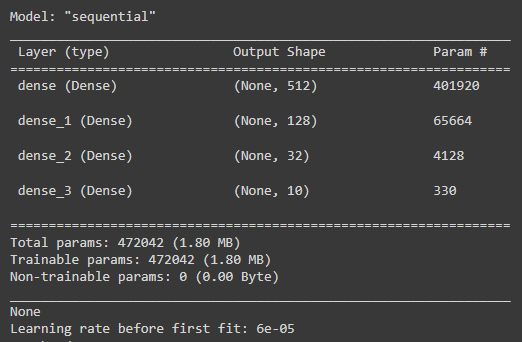
## ۱-۲. **آشنایی با مجموعه دادگان**

* مجموعه دادگان MNIST مجموعه‌ای از تصاویر 28\*28 است که در مجموعه آموزش 60 هزار و در مجموعه آزمون 10 هزار داده وجود دارد.
* در شکل 3 نمونه‌ای از هر کلاس دیده می‌شود.
* هیستوگرام مربوط در شکل 4 دیده می‌شود. داده‌ها تاحدودی توزیع یکنواخت دارند. مجموعه دادگان MNIST تا حد خوبی بالانس است و نیاز به پردازش خاصی ندارد. تنها داده‌ها را قبل از آموزش و آزمون Reshape و نرمالایز و همچنین لیبل‌ها را انکود One-Hot می‌کنیم.
* برای بهبود همگرایی، جلوگیری از Vanishing Gradients، بهبود تعمیم پذیری، تضمین پایداری و...

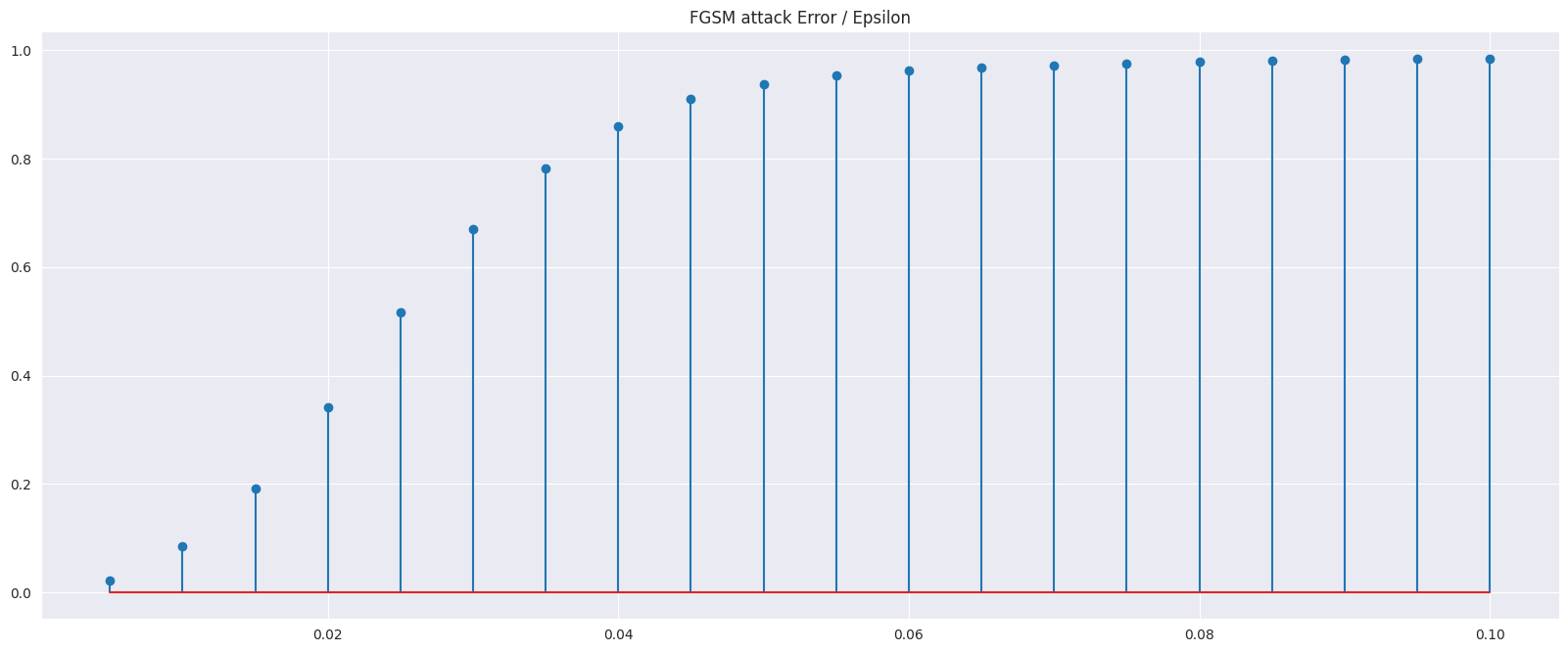
## 2-2. ایجاد و آموزش مدل

در این بخش از کتابخوانه Keras برای ایجاد و آموزش مدل استفاده می‌کنیم. مشخصات مدل پیاده شده در شکل 5 مشاهده می‌شود.

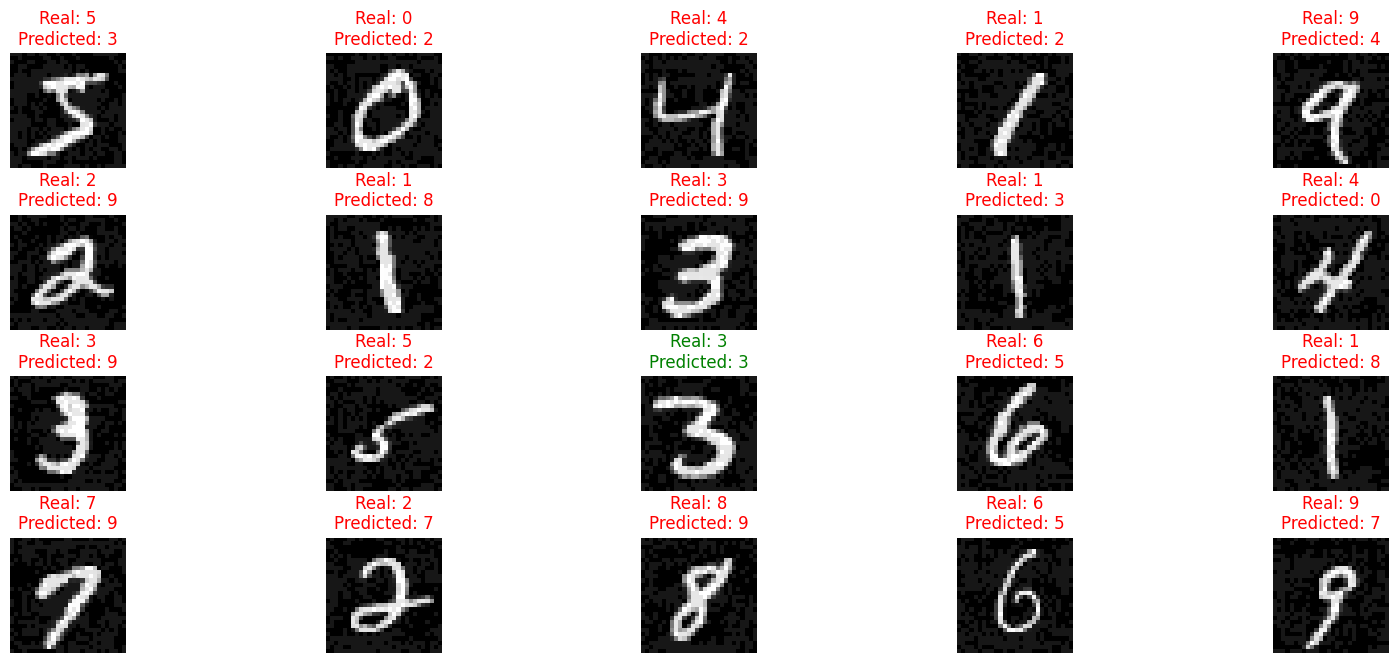
* همانطور که در بخش قبل گفته شد، داده‌ها را قبل از آموزش و آزمون Reshape می‌کنیم. به این دلیل هر داده 28\*28 تبدیل به یک بردار 1\*784 می‌شود.
* در مسائل کلاس‌بندی با بیش از دو کلاس معمولا از تابع Softmax استفاده می‌کنیم. این تابع خروجی‌های مدل (logits) را به یک توزیع احتمالاتی در کلاس‌های خروجی تبدیل می‌کند و در نتیجه به هر کلاس یک احتمال نسبت می‌دهد، خروجی را نرمالیزه می‌کند و همچنین تابعی مشتق پذیر است.



شکل 5. مشخصات مدل پیاده شده



شکل 6. نمودار خطا بر اساس پارامتر epsilon در حمله FGSM



شکل 7. نمونه داده‌ها و لیبل واقعی و خروجی مدل در حمله FGSM

## 3-2. پیاده‌سازی حمله FGSM

در پیاده‌سازی این حمله از کتابخانه tensorflow کمک گرفته‌ایم.

* کد پیاده‌سازی در فایل نوت‌بوک موجود است.
* نمودار خطای خروجی بر اساس تغییرات پارامتر epsilon با گام‌های 0.005 در شکل 6 مشاهده می‌شود. واضح است که با افزایش مقدار epsilon خطا افزایش می‌یابد و در نهایت تقریبا به 100 درصد نمونه‌ها خواهد رسید. بدیهی است هرچه مقدار بیشتر شود اثر آن به طور ظاهری توسط چشم نیز در نمونه‌ها دیده می‌شود.
* نمونه‌ای از داده‌ها با نویز و لیبل اصلی و پیشبینی مدل از آنها در شکل 7 رسم شده است.

جدول 2. نتایج حمله PGD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Epsilon** | **Alpha** | **Iteration** | **Error (%)** |
| **آزمایش 1** | 0.01 | 0.001 | 10 | 86.9 |
| **آزمایش 2** | 0.01 | 0.001 | 40 | 90.10 |
| **آزمایش 3** | 0.1 | 0.001 | 10 | 90.05 |
| **آزمایش 4** | 0.1 | 0.005 | 10 | 98.455 |
| **آزمایش 5** | 0.1 | 0.001 | 15 | 90.655 |
| **آزمایش 6** | 0.1 | 0.001 | 20 | 92.04 |
| **آزمایش 7** | 0.1 | 0.001 | 40 | 97.83 |
| **آزمایش 8** | 0.1 | 0.005 | 10 | 98.455 |
| **آزمایش 9** | 0.1 | 0.0015 | 15 | 92.91 |

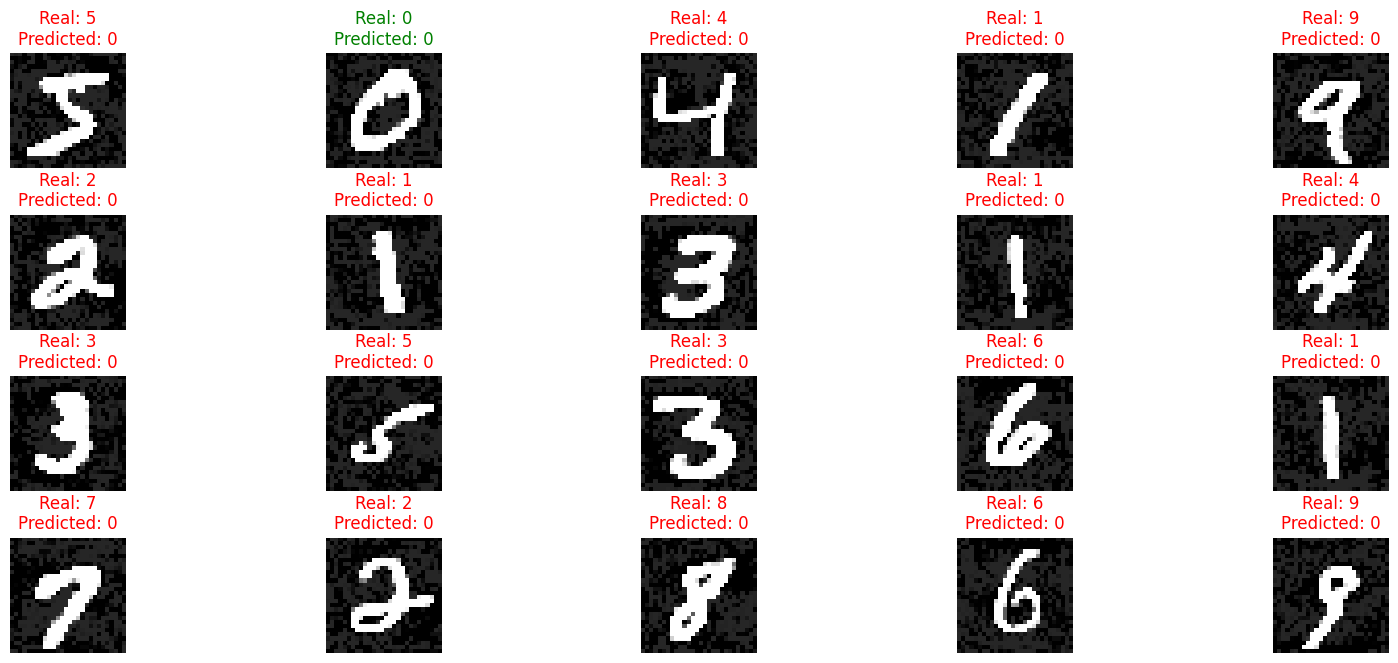
## 4-2. پیاده‌سازی حمله PGD

برای این حمله نیز از tensorflow بهره می‌گیریم. این حمله از حمله قبل سنگین‌تر است در نتیجه برای تحلیل پارامترها را در چند آزمایش تغییر داده و خطا را حساب می‌کنیم. نتیجه آزمایش‌ها در جدول 1 آورده شده است. نتایج به شرح زیر است:

* با تغییر پارامتر epsilon و ثابت نگه داشتن بقیه پارامترها در این حمله در تصاویر از نظر ظاهری تغییر کمی دیده می‌شود اما همانطور که در آزمایش 1 و 3 مشاهده می‌شود خطا و اثر بخشی حمله را بیشتر می‌کند.
* تغییر پارامتر alpha با ثابت نگه داشتن پارامترهای دیگر، در آزمایش 3 و 4، علاوه بر اینکه خطا را بیشتر می‌کند در ظاهر نیز تصاویر را خراب می‌کند.
* پارامتر Iteration مشابه alpha با زیاد شدن خطا را زیاد و ظاهر تصاویر را خراب‌تر می‌کند. افزایش این پارامتر به طور قابل توجهی زمان حمله را افزایش می‌دهد.

پاسخ به سوالات:

* پیاده‌سازی در نوت‌بوک سوال دوم موجود است.



شکل 8. نمونه‌های آزمایش 5 و حمله PGD

* این روش بر اساس گرادیان تابع تلفات نسبت به ورودی و انجام مکرر (Iterations) این محاسبات روی داده ورودی انجام می‌شود و همچنین سعی می‌کند خطا را در بازه خاصی نگه دارد. در روش FGSM که تکرار نداریم و در یک مرحله انجام می‌شود، حمله بر اساس علامت تابع گرادیان تلفات روی داده ورودی خواهد بود. به همین دلیل چون در یک مرحله انجام می‌شود از نظر محاسباتی سبک‌تر است اما حمله PGD حمله بالقوه قوی‌تری است و شناسایی و مقابله با آن می‌تواند برای شبکه عصبی سخت‌تر باشد.
* همانطور که گفته شد حمله بالقوه قوی‌تری است، پارامترهای بیشتر و درجه آزادی بیشتری به حمله‌کننده می‌دهد. و شناسایی و مقابله با آن برای شبکه عصبی می‌تواند دشوار باشد. در این آزمایش مشاهده شد که با این حمله می‌توان با حفظ ظاهر نمونه‌ها به خطای بیشتری رسید.
* نمونه از هرکلاس، لیبل اصلی و لیبل پیشبینی شده توسط مدل در شکل 8 ارائه شده است.

# **پرسش ۳** **– عنوان پرسش سوم به فارسی**

## ۱-۳. **عنوان بخش اول**

متن نمونه

.

# **پرسش ۴** **– عنوان پرسش چهارم به فارسی**

## ۱-۴. **عنوان بخش اول**

متن نمونه

.