FGSM – Detektering och Skydd

Sebastian Bengtsson, sebe20@student.bth.se, Marius Stokkedal, matk20@student.bth.se

***Sammanfattning*—Denna rapport är ett vidare arbete av rapporten [2]. Arbetet är till för att utveckla ett försvar för bilklassificerare av flygfoton – i detta fallet en Convolutional neural network (CNN) modell. Attackmodellen som används är Fast Gradient Sign Method (FGSM) och modellerna som används för försvar är: Support Vector Machine (SVM), k-Nearest Neighbors (KNN) och Decision Tree (DT). Det första försöket till skyddsmodeller funkar inte som hoppats – den mesta av datan klassas som att den är attackerad oavsett om den är det eller inte. Uppdelning av datan producerar mer lovande resultat, men det undersöks endast som ett koncept och för att bygga en grund för framtida forskning.**

I. Introduktion

FGSM är en effektiv typ av attack mot AI-modeller. I dess natur ingår att attacken ska vara svår att upptäcka, både för skyddsmodeller och med blotta ögat. Detta arbete är en fortsättning på rapporten [2]. Målet i denna rapport är att utforska fler alternativ för att skydda ett CNN från attacker utförda med Fast Gradient Method (FGSM) med hjälp av modellerna Support Vector Machine (SVM), KNN (k-Nearest Neighbor) och DT (Decision Tree).

II. Metod

Författarna till [2] skriver vilka dataset som används men de delar inte med sig av sin kod; alltså påbörjades arbetet genom att föresöka återskapa modellen från *SECTION III. Methodology*.

A. Datan

För att utgå från samma klasser som används i [2] användes samma källa – [5]. Dataseten kommer från [3] och [4]. Klasserna som används är:

a. Building (Byggnad)
b. Avenue (Aveny)
c. Road (Väg)
d. Airport (Flygplats)
e. Storeroom (Silo)
f. Roadside tree (Träd vid sidan av väg)
g. Residents (Bostadsområde)
h. Bridge (Bro)
i. Parking lot (Parkering)
j. Marina (Småbåtshamn).

När datan hade laddats fanns inte klassen **f** – *Roadside tree (Träd vid sidan av väg)*. De klasser som inte används i [2] togs bort från datan. För att kombinera de olika filerna gjordes antaganden baserat på namnen på klasserna och visuell inspektion av bilderna i de fall det var oklart om klasserna är samma. Om de var samma kombinerades bilderna till samma mapp; annars placerades de i separata mappar med unika namn för att de skulle kunna skiljas åt. De klasser som används i detta arbete är:

a. Building (Byggnad)
b. Avenue (Aveny)
c. Road (Väg)
d. Airport (Flygplats)
e. Storeroom (Silo)
f. Denseresidential (Tätbebygt bostadsområde)
g. Mediumresidential (Medelbebtggt bostadsområde)
h. Residents (Bostadsområde)
i. Bridge (Bro)
j. Parking lot (Parkering)
k. Marina (Småbåtshamn).

Bilderna från [3] och [4] är 256x256 pixlar stora men i [2] är bilderna 128x128 pixlar. För att återskapa ett så likt resultat som möjligt, spara lagringsutrymme och minska träningstiden krymptes storleken på alla bilder efter att de hade laddats ner till 128x128 pixlar.

B. Återskapa CNN

I [2] framgår det inte tydligt hur deras CNN är uppbyggt, men eftersom att denna inte är en central del i själva arbetet skapades ett generiskt CNN för att ha något att utgå ifrån. Strukturen bygger till stor del på den TensorFlows exempel från deras dokumentation [6]. CNN-modellen i detta arbeta har en träffsäkerhet på cirka 98% och den tränades med den tidigare nämnda datan. Datan, på 22 256 bilder, delades upp i 80% träningsdata och 20% valideringsdata; samma uppdelning som i [2].

C. FGSM

FGSM används för att attackera bilder med störningar och på så sätt lura CNN så det missklassifierar bilden. FGSM är snabbt och billig beräkningsmässigt vilket gjorde det till en passande attack eftersom största fokuset låg på försvarsdelen.

Attacken skapas genom att först räkna ut gradienterna av CNNs loss vilket ger riktningen där loss-en ökar snabbast. Tecknet (plus eller minus) används för att se till att störningen i bilden är liten och är i samma riktning som gradienten. Det multipliceras med ett litet tal (epsilon) för att skapa störningen. Ett antal olika epsilon har testats (0.01, 0.05 och 0.1), större värden på epsilon gör störningen mer påverkande men också mer upptäckbar.

D. Extrahera attribut från CNN-modellen

I [2] extraherar de attributen från det första och andra fullt sammanlänkade lagret och sedan används detta för att träna skyddsmodellen. Modellen tränas med attribut från bara lager ett, bara lager två och lager ett och två sammansmälta. För denna delen av arbetet används både rena bilder från träningsdatan samt de bilder som attackerats med FGSM.

E. Uppdelning av datan

Två olika uppdelningar görs, en där det hela rena datasetet används tillsammans med hela datasetet attackerat med FGSM på epsilon; 0.01, 0.05 och 0.1. Detta ger en obalans, där 3/4 är attackerade bilder samt att skillnaderna mellan de rena och 0.01 epsilon bilderna är väldigt små vilket senare visar sig göra det svårt för modellerna. Så den andra gruppen blir rena bilder på airport klassen, samt attackerade bilder i airport klassen med epsilon 0.1 för att visa att det går att skilja på attackerade och icke-attackerade bilder.

III. Modeller

Försvarsmodellerna som används för att klassificera bilder som attackerade eller inte attackerade använder sig utav 22 256 features. Dessa blandas först och sedan delas de upp i tränings- och testdata med en uppdelning på 80% respektive 20%; samma uppdelning som i [2]. Datan bestod av 4 olika typer av bilder: orginalbilderna som inte var attackerade och bilder attackerade med FGSM med ett epsilon värde på 0.1; 0.05 och 0.01.

A. SVM (Support Vector Machine)

SVM fungerar genom att skapa ett hyperplan eller en uppsättning hyperplan i en högdimensionell rymd, ett plan som bäst delar upp klasserna. Den är robust mot overfitting, särskilt i högdimensionella rymder. Denna modell är den samma som används i [2].

B. KNN (k-Nearest Neighbors)

KNNs idé är att klassificera en ny punkt baserat på likheten med 'k' närmaste grannar. Dess enkelhet och förmåga att träna modellen med lite beräkning gjorde den till ett likgiltigt val med SVM, träningstiden var snabbare och modellens filstorlek var mindre än SVM. Olika k-värden testades och 5 valdes till slut.

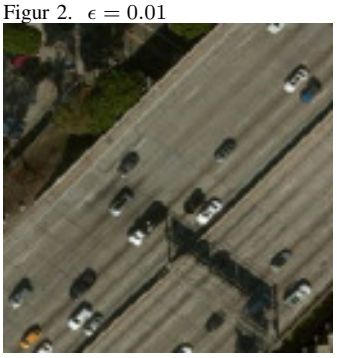
C. DT (Decision Trees)

DTs fattar beslut genom att dela upp datamängden i allt mindre undergrupper baserat på olika kriterier. Modellen hade snabbast träningstid och minst filstorlek.

IV. Resultat

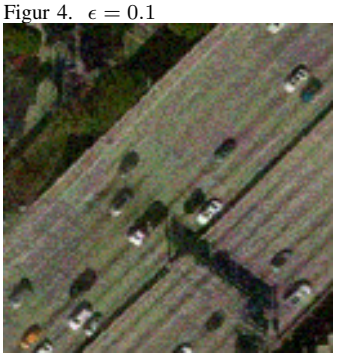
Figur 1. Clean image

Figur 2. $\epsilon = 0.01$



Figur 3. $\epsilon = 0.05$

Figur 4. $\epsilon = 0.1$



I figur 1 ses en exempelbild innan attack och i figur 2-4 syns samma bild efter FGSM-attacken med olika epsilonvärden. I figur 2 är det praktiskt taget omöjligt att se en attack med blotta ögat medans i figur 3 och 4 kan mer tydligt och ännu mer tydligt se ett avvikande brus, speciellt längs vägen.

Category	**Clean**	$\epsilon = 0.01$	$\epsilon = 0.05$	$\epsilon = 0.1$
Airport	98.84%	89.2%	87.79%	75.19%
Avenue	92.1%	92.46%	93.01%	94.12%
Bridge	92.96%	92.32%	78.68%	35.39%
Building	69.0%	71.0%	73.0%	67.0%
Denseresidential	70.0%	66.0%	62.0%	62.0%
Highway	96.41%	92.83%	85.2%	50.67%
Marina	95.28%	96.14%	93.78%	41.2%
Mediumresidential	58.0%	32.0%	31.0%	29.0%
Parkinglot	97.88%	97.18%	95.24%	88.89%
Residents	99.51%	99.75%	99.63%	98.64%
Storeroom	98.08%	98.15%	95.59%	86.57%
Total	**95.31%**	**93.35%**	**90.64%**	**76.33%**

Tabell I
Accuracy efter FGSM-attack

Tabell 1 visar hur väl de olika kategorierna klassificerades av CNN-modellen i utgångsläget ("Clean") samt efter störningar med olika epsilon-värden ("ϵ = 0.01", "ϵ = 0.05", "ϵ = 0.1"). Överlag så går alla kategorierna ner efter ökad störning. Mediumresidental tar sig ett rejält fall direkt men hade redan från början låg träffsäkerhet. Anmärkningsvärt är även Avenue som istället går upp i träffsäkerhet ju mer epsilon ökar.

När ϵ = 0.01 så var det en ytterst liten sänkning av accuracyn något väntat. Det sänkte dock accuracyn, utan att attacken tydligt syntes att på bilderna.

När ϵ = 0.05 blev attacken tydligare och totalaccuracyn hade då sjunkit med cirka 5 procentenheter (och procent) från innan

attacken. Kategorin bridge blev särskilt träffad och gick från 92.96% till 78.68%.

När $\epsilon = 0.1$ är attacken mycket tydlig, men också skapas ordentlig skada, totala accuracyn går ner till 76.33% och kategorin bridge faller nu ner till endast 35.39%.

En graf med accuracyn på y-axeln och värdet på ϵ på x-axeln ger att K-värdet (lutningen) mellan 0.01 och 0.05 var på -.934 medans mellan 0.05 och 0.1 var det -4.67. Detta visar att attackskadan inte ökar linjärt utan det går fortare desto större värde på ϵ, men skillnaden i störningen var också tydligare för ögat mellan 0.05 och 0.1 än vad det var mellan 0.01 och 0.05.

A. *Försvarsmodeller*

Metric	SVM	Decision Tree	KNN
Accuracy	74.57%	59.75%	71.88%

Tabell II
RESULTATET FÖR SKYDDS-MODELLERNA

Resultatet för alla modellerna när de tränades på all datan. Referensvärdet från [2] är cirka 90% för SVM.

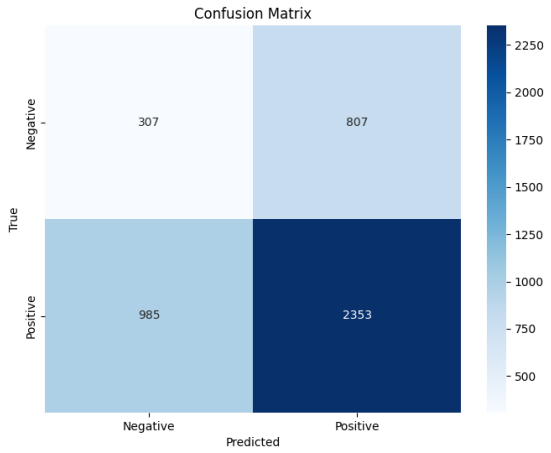
Metric	SVM	Decision Tree	KNN
Accuracy	84.94%	75.96%	85.58%

Tabell III
RESULTATET FÖR SKYDDS-MODELLERNA TRÄNADE PÅ ENDAST *airport*

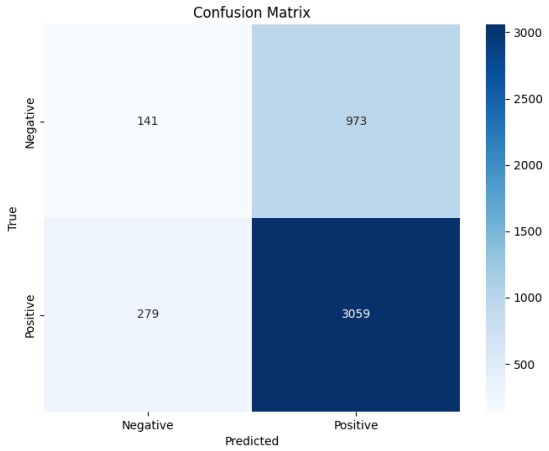
Resultatet för alla modellller när de endast tränas på rena bilder och bilder attackerade med ett epsilonvärde på 0.1 där alla bilder endast kommer från klassen *airport*.

B. *Confusion matrixes*

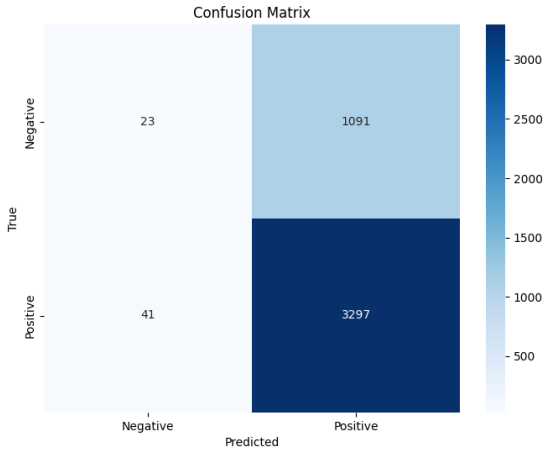
Nedan kommer confusion matrixes för samtliga skyddsmodeller. De som är markerade med *"version 2"* i sin bildtext är de som är tränade på rena bilder och bilder attackerade med ett epsilonvärde på 0.1 där alla bilder endast kommer från klassen *airport*.



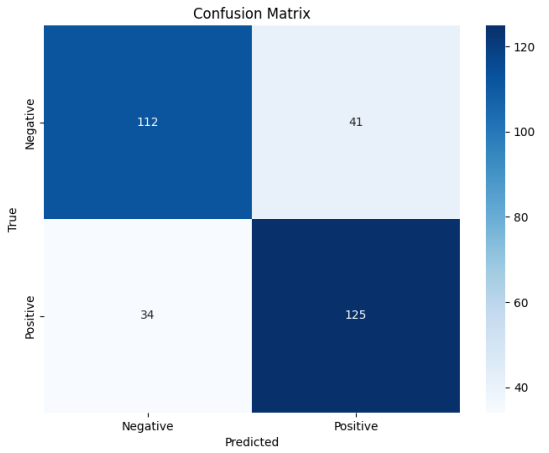
Figur 5. Confusion matrix för Decision Tree



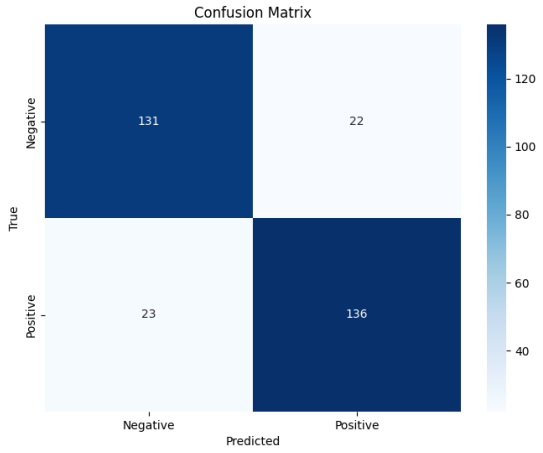
Figur 6. Confusion matrix för KNN



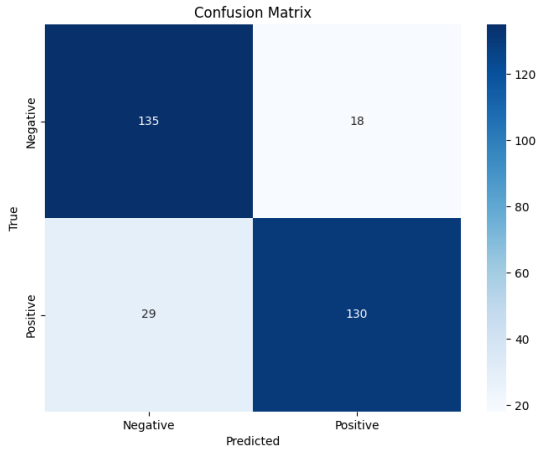
Figur 7. Confusion matrix för SVM



Figur 8. Confusion matrix för Decision Tree version 2



Figur 9. Confusion matrix för KNN version 2



Figur 10. Confusion matrix för SVM version2

Samtliga confusion matrixes visar ett tydligt mönster: att träna på endast mer attackerad data och klassvis ger ett bättre resultat baserat på True Positive, True Negative, False Positive och False Negative. Kombineras matriserna med tabell II samt tabell III blir denna trend ännu tydligare där träffsäkerheten för modellerna också blir betydligt bättre med version 2.

V. Diskussion

A. FGSM

Resultaten från Tabell 1 visar hur det gick för CNN-modellen att klassificera bilderna innan och efter de blivit attackerade med FGSM-störningar. Innan attackerna gick det överlag bra för CNN att klassificera, alla kategorierna lyckadesdes träffa rätt på över 90% förutom 3, building, dense residental och medium residential. Detta förklaras genom att dessa 3 kategorier hade mycket mindre antal träningsdata, endast 100 av varje kategori istället för cirka 800 som var snittantalet annars. Man ser det också när accuracyn för de 3 sänks avsevärt mer efter attackerna jämfört med de andra kategorierna.

B. Datan och träning av modellerna

Efter att modellerna tränats endast rena bilder och FGSM-attackerade bilder med ett epsilonvärde på 0.1 och bilder bara från klassen *airport* blev resultatet betydligt bättre än när modellerna tränades på all datan. Detta återspeglas både träffsäkerheten och fördelningen av true positive, false positive, true negative och false negative. Grundidén att klumpa ihop all data utan att skilja dem klassvis var alltså sämre än att träna modellerna på de olika klasserna på bilderna, men detta koncept är lättare att genomföra i teorin än i praktiken. Poängen med FGSM är att bilderna ska klasssas som en felaktig klass, alltså hade man inte kunnat använda sig utav klassen på bilden i själva klassificeringssteget. Dock är det en intressant tanke och något som är värt att undersöka vidare.

VI. Slutsatser

För FGSM-attacken så hade ett optimalt värde på epsilon kanske kunnat ligga mellan 0.1 och 0.05, där attacken gör så mycket som möjlig skada och är så diskret som möjlig.

Det är möjligt att strukturen på CNN-modellen påverkar hur lätt den är att skydda. Det lagret från modellen som användes har 128 noder, för att få mer data att jobba med skulle möjligtvis detta kunnat ökas till ett större antal. Försöken att försöka återskapa tog mycket mer tid än väntat så vissa experiment som hade planerats fick tas bort. Dock finns det nnu är färdig grund att bygga på så detta kanske blir ett jobb i framtiden.

Att använda sig av ett extraherat lager från en CNN-modell som i detta fallet har varit ett spännande och lärorikt arbete, men för att detta arbetet ska kunna vara applicerbart hade det behövts mer tid.

VII. Vidare forskning

Som sagt hade det varit intressant att undersöka olika strukturer på CNN-modellen vidare samt att försöka få in klassen på bilden i datan som används vid klassificering på ett bra sätt. Skyddsmodellerna hade behövt mer data att gå på än bara det från CNN-modellens lager, att ta fram detta hade möjligtvis hjälpt klassificeringen och hade varit ett intressant spår att fortsätta på i framtiden.

Referenser

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] W. Li et al., Spear and Shield: Attack and Detection for CNN-Based High Spatial Resolution Remote Sensing Images Identification,"in IEEE Access, vol. 7, pp. 94583-94592, 2019, doi: 10.1109/ACCESS.2019.2927376.

[3] Yi Yang and Shawn Newsam, Bag-Of-Visual-Words and Spatial Extensions for Land-Use Classification,ÄCM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS), 2010.

[4] Li, H.; Dou, X.; Tao, C.; Wu, Z.; Chen, J.; Peng, J.; Deng, M.; Zhao, L. RSI-CB: A Large-Scale Remote Sensing Image Classification Benchmark Using Crowdsourced Data. Sensors 2020, 20, 1594. https://doi.org/10.3390/s20061594

[5] W. Li, H. Liu, Y. Wang, Z. Li, Y. Jia and G. Gui, Deep Learning-Based Classification Methods for Remote Sensing Images in Urban Built-Up Areas,"in *IEEE Access*, vol. 7, pp. 36274-36284, 2019, doi: 10.1109/ACCESS.2019.2903127.

[6] TensorFlow, "Convolutional Neural Network (CNN)", *TensorFlow*, 2019. https://www.tensorflow.org/tutorials/images/cnn