# DNN for deep-fake recognition

Francesco Cozzuto, Emanuele d'Ajello, Marco D'Elia, Raffaele del Gaudio
Progetto ESM - gruppo 15
(a.a. 2020-2021)

## 1 - Introduzione

Uno dei problemi emergenti del 21° secolo nel panorama della disinformazione online è la falsificazione video con utilizzo di tecniche di Intelligenza Artificiale, in particolare con le "Deep Neural Network" (DNN).

Una particolare categoria di video manipolati, comunemente conosciuta come "deep-fake", ha raggiunto uno sconcertante grado di perfezionamento e tali video risultano in molti casi indistinguibili dagli originali.
Un deep-fake è un video in cui la faccia di un individuo target è sostituita da quella di uno donatore generata da una DNN, mantenendo le espressioni facciali e la posizione della testa dell'individuo target.

Risulta chiaro il perché della grande preoccupazione che deriva questo fenomeno. Basti pensare a tutte le implicazioni che una falsa dichiarazione fatta in video da una persona influente potrebbe scaturire sul mondo.

Il nostro lavoro si prefigge l'obiettivo di realizzare, proprio attraverso le tecnologie delle DNN, un meccanismo di riconoscimento dei deep-fake.

Nel nostro lavoro si è partiti dal dataset Celeb-DF [1]. Tuttavia è stato necessario elaborarlo per estrapolare i frame dai video e ritagliare i volti. In questa fase ci siamo avvalsi di Retina-Face, un noto software di Face Localisation.

## 2 - Approccio implementato

Il problema ha richiesto come approccio di risoluzione l'addestramento di una rete neurale che distingua video manipolati dagli altri. Per giudicare la veridicità di un video, vengono estratti dei volti dai suoi frame ed analizzati individualmente dalla rete. Se le immagini estrapolate risultano manipolate nel loro insieme, allora il video è ritenuto anch'esso manipolato.

### 2.1 - Architettura

L'architettura usata, mostrata in Figura 1, è ottenuta mettendo in serie a ResNet-50 due livelli densi.
L'istanza di ResNet-50 usata è stata pre-allenata sul dataset ImageNet, quindi il problema si è trasformato in uno di fine-tuning.
La rete è strutturata in modo da ricevere in input delle immagini RGB con dimensioni 224x224.

Il primo livello denso è composto da 3000 neuroni ed impiega come funzione di attivazione Leaky-ReLU (con pendenza 0.05). Dato che il problema è di classificazione binaria esclusiva (ogni immagini può appartenere alla classe di immagini contraffatte e di non contraffatte, ma non allo stesso tempo), il secondo livello di output ha 2 neuroni e funzione di attivazione SoftMax.

### 2.2 - Addestramento

Per allenare la rete, il dataset di video Celeb-DF è stato scomposto in un dataset di immagini statiche. Questo processo è approfondito nella sezione relativa al dataset.

La funzione di quantificazione dell'errore usata durante il training è la *Categorical Cross Entropy*, mentre l'ottimizzatore usato è *Adam*.

L'efficacia massima del training è stata raggiunta per learning rate di $10^{-4}$e 15 livelli bloccati di ResNet-50. Per problemi tecnici non è stato possibile aumentare il batch size oltre i 64 elementi. L'architettura ha raggiunto il picco delle performance dopo 3 epoche di fine-tuning.

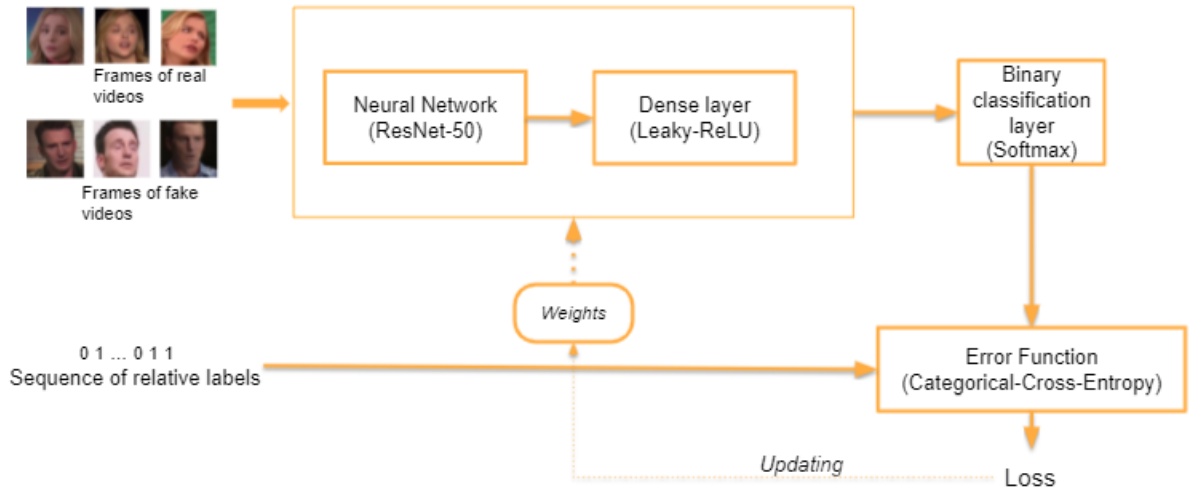


Figura 1: *Schema a blocchi dell'implementazione della rete neurale.*

## 3 - Dataset

In questo lavoro si è partiti dal dataset Celeb-DF, che è una collezione di 5639 video di deep-fake e di 590 video non artefatti. Vale la pena notare la qualità di questo dataset rispetto alle alternative. Per evidenziare ciò; in Figura 2 sono messi a confronto due frame estratti rispettivamente da Celeb-DF e da Google Deep-Fake detection dataset,entrambi manipolati, è evidente come il primo frame sia molto più realistico del secondo..

Dato che la rete prende come input immagini, a partire dal dataset di video è stato generato un dataset di volti. Da ogni video artefatto sono stati estratti 10 volti, mentre dagli altri sono stati estratti 100 volti. Questo processo è stato possibile grazie a Retina-Face, un software di *face localization*. Per non perdere varietà durante la fase di campionamento dei video, si è avuto cura di massimizzare il passo di campionamento, ossia la distanza di ciascun video estratto l'uno dall'altro,come si può osservare a Figura 3 e Figura 4.

Figura 2: *Entrambe queste immagini sono state manipolate: a sinistra il volto fornito fornito da Celeb-DF, mentre a destra dal dataset di Google.*

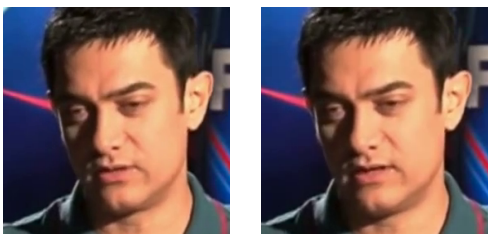

Figura 3: *Volti estratti da video con passo di campionamento pari ad 1, ossia in frame consecutivi.*

Figura 4: *Volti estratti con passo di campionamento massimo.*

## 4 - Risultati sperimentali

In questa sezione saranno esaminati i risultati ottenuti studiando le prestazioni di varie architetture, ottenute variando il numero di livelli densi, il numero di neuroni che compongono i livelli densi nascosti ed in fine il numero di livelli bloccati di Resnet50 durante la fase di training.

### 4.1 - Rete con un livello denso

Le prima architettura allenata è stata quella con un singolo livello denso posto dopo i livelli di Resnet50. I seguenti risultati sono stati ottenuti su un training durato 3 epoche ed al variare dei livelli bloccati di Resnet50, ossia per 5, 10, 15 e 20 livelli bloccati. Se non specificato altrimenti, il learning rate è stato fissato a $10^{-4}$, mentre il batch size a 64 elementi.

Le prestazioni sul dataset di test al crescere delle epoche sono state strettamente crescenti, indipendentemente dal numero di livelli bloccati. Tuttavia per 15 e 20 livelli bloccati sono state nettamente migliori.

| Livelli bloccati di Resnet50 | Loss dopo 3 epoche (Test) | Accuracy dopo 3 epoche (Test) |
|---|---|---|
| 5 | 65.10 | 0.51 |
| 10 | 1.26 | 0.48 |
| 15 | 0.68 | 0.84 |
| 20 | 0.49 | 0.84 |

Inoltre è possibile osservare come il valore di loss ottenuto per 5 livelli bloccati sia molto più grande degli altri, di 2 ordini di grandezza rispetto a quelli ottenuti per 15 e 20 livelli bloccati. Considerato che invece in training la rete con 5 livelli bloccati sia salita da un'accuratezza di 0.96 a 0.99, ci permette di trovare una giustificazione di questo comportamento nel fatto che aumentando i livelli bloccati durante il training, la rete si presti di più ad adeguarsi ai dati di training, a discapito di quelli di test.

### 4.2 - Rete con un livello denso nascosto composto da 1000 neuroni

Il secondo set di architetture che sono state testate sono quelle con due livelli densi, studiate al variare del numero di neuroni del livello denso nascosto. Se non specificato altrimenti, la funzione di attivazione impiegata per il livello nascosto è ReLU.

I risultati ottenuti dopo 3 epoche di allenamento sono simili a quelli ottenuti per la rete con un unico livello denso:

| Livelli bloccati di Resnet50 | Loss dopo 3 epoche (Test) | Accuracy dopo 3 epoche (Test) |
|---|---|---|
| 5 | 165.20 | 0.49 |
| 10 | 2.10 | 0.51 |
| 15 | 0.26 | 0.93 |
| 20 | 0.50 | 0.82 |

Per 5, 10 livelli bloccati le prestazioni sono decisamente inferiori a quelle ottenute con 15, 20 livelli. Inoltre, il valore di loss relativa all'allenamento con 5 livelli bloccati è, di nuovo, sproporzionatamente grande rispetto agli altri casi. Per 5 livelli il valore di loss ottenuto alla prima epoca era di solo 13.83. Questo porta a pensare che un learning rate di $10^{-4}$sia troppo grande. Abbassandolo a $10^{-5}$ si ovvia al problema ottenendo loss ed accuracy in test di, rispettivamente, 2.52 e 0.50 dopo 3 epoche. Che comunque sono meno interessanti di quelli ottenuti per 15, 20 livelli bloccati.

La rete ottenuta dopo 3 epoche per 15 livelli bloccati è un risultato notevole per questa architettura. Proseguendo l'allenamento oltre le 3 epoche le prestazioni in training crescono mentre quelle in test degradano, ossia c'è over-fitting.

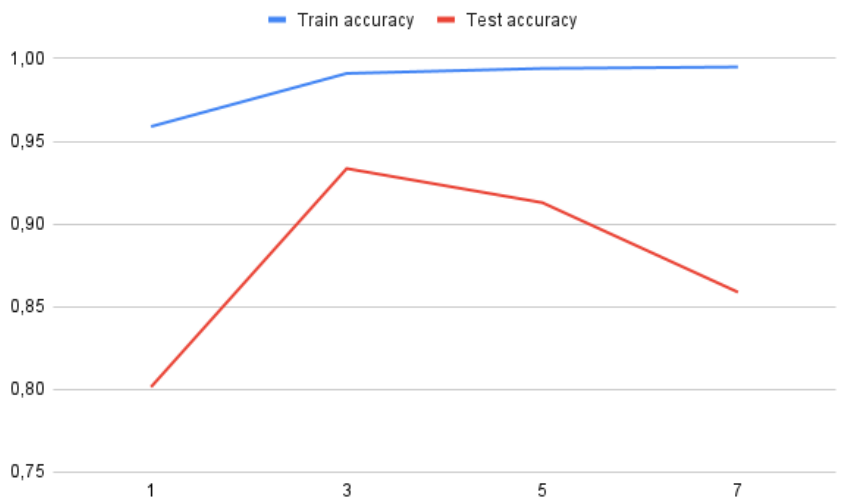


Figura 5*: Accuracy della rete con un livello nascosto da 1000 neuroni all'aumentare delle epoche di allenamento.*

### 4.3 - Rete con un livello denso nascosto composto da 2000 neuroni

Aumentando il numero di neuroni del livello denso nascosto a 2000, dopo 3 epoche di allenamento, si sono ottenuti i risultati:

| Livelli bloccati di Resnet50 | Loss dopo 3 epoche (Test) | Accuracy dopo 3 epoche (Test) |
|---|---|---|

| | | |
|---|---|---|
| 15 | 0.19 | 0.94 |
| 20 | 0.45 | 0.85 |

Si è ritenuto che il numero adeguato di livelli bloccati durante il training fosse 15 o 20 poiché la variazione del numero di neuroni del livello nascosto non rappresenta un cambiamento sufficientemente significativo per cambiare i trend che si sono presentati per 1000 neuroni.

Ancora, la rete con 15 livelli bloccati è superiore a quella con 20. Tuttavia per 3 epoche si ha il rendimento massimo. Allenarla ulteriormente la porta in over-fitting.

### 4.4 - Rete con un livello denso nascosto composto da 3000 neuroni

Aumentando il numero di neuroni a 3000 per la rete con 15 livelli bloccati il rendimento in fase di test non supera quella con 2000 neuroni, ma converge ad un'accuratezza compresa tra 0.92 e 0.93, come mostrato in Figura 6.

Questo comportamento è stato attribuito al fatto che la funzione di attivazione ReLU sia nulla per input negativi. In certe condizioni è possibile che una buona parte della rete si trovi ad operare nella sua regione nulla, riducendo l'efficacia dell'algoritmo di correzione dei pesi. Per ovviare a questo problema abbiamo sostituito la funzione di attivazione ReLU con la funzione Leaky-ReLU (con pendenza 0.05). Questo ci ha permesso un ulteriore, nonché più rapido, miglioramento.

| Epoche | Loss (Test) | Accuracy (Test) |
|---|---|---|
| 1 | 0.3152 | 0.895 |
| 2 | 0.2508 | 0.9206 |
| 3 | 0.1645 | 0.9463 |

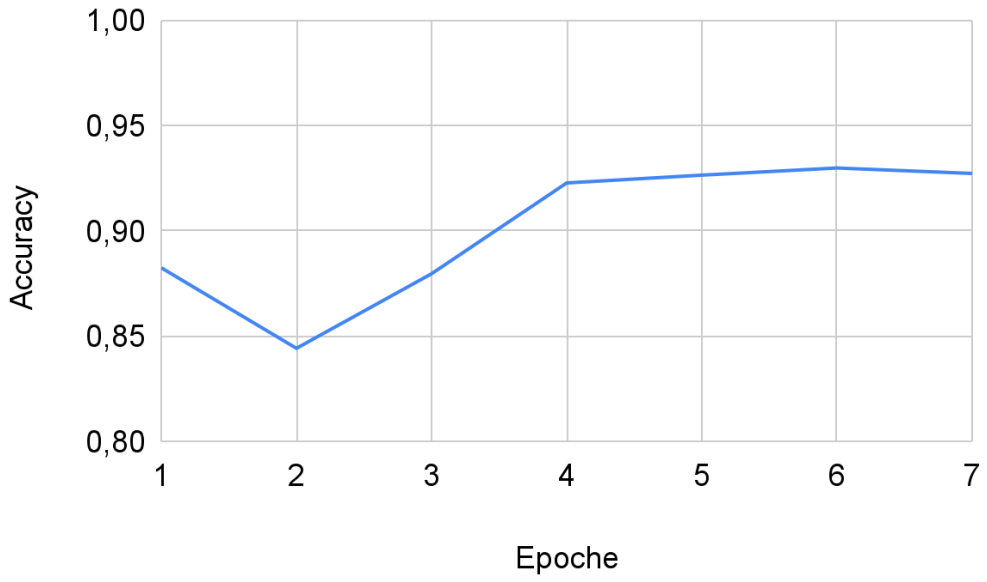


Figura 6*: Accuracy in test della rete con un livello nascosto da 3000 neuroni ed attivazione ReLU che converge tra 0.92 e 0.93.*

### 4.5 - Efficacia del dataset con passo di campionamento massimo

Per sfidare l'assunzione secondo la quale generare il dataset di frame a partire da quello di video con passo di campionamento maggiore fosse meglio, è stata riallenata la rete migliore ottenuta, sul dataset di volti con passo di campionamento minimo pari ad 1. Diminuendo il passo di campionamento la rete va immediatamente in over-fitting. Il seguente grafico confronta l'accuracy in train ed in test della rete allenata sui due dataset. È evidente come, a parità di rendimento in training, la rete allenata sulle immagini a varianza minore fra di loro perda subito di accuratezza in test.

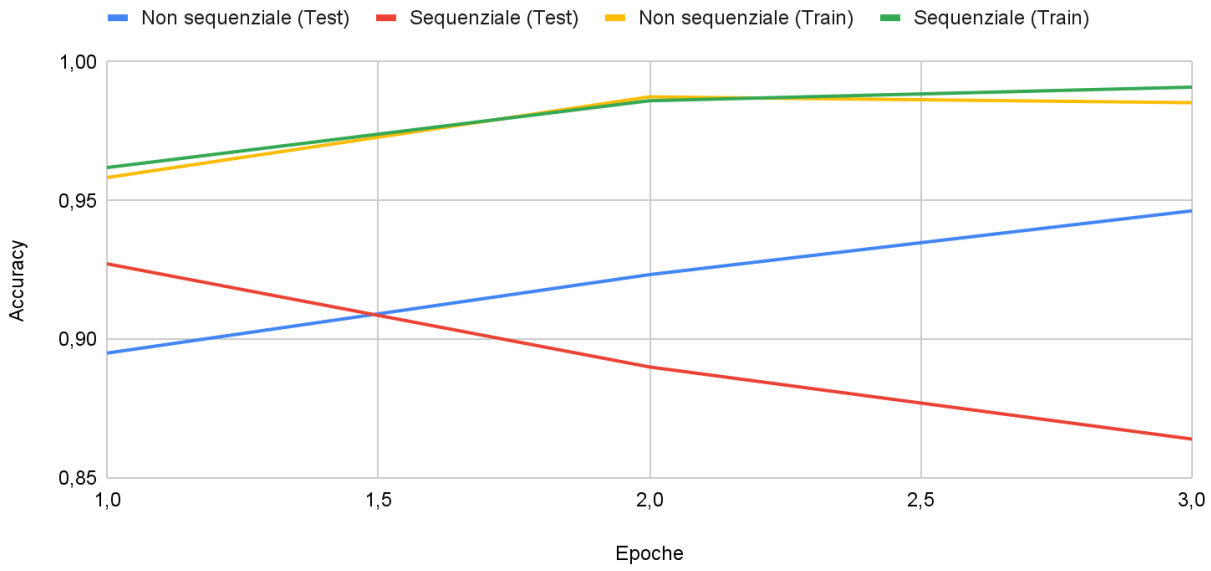


Figura 7: *Accuratezza della rete con due livelli densi ed attivazione Leaky-ReLU in quello intermedio allenata sul dataset generato con passo di campionamento pari ad 1 (sequenziale) e con passo di campionamento massimo (non sequenziale).*

## 5 - Attacco alla DNN

Una volta creato il modello si è sperimentato un attacco avversario.
L'attacco consiste nel sommare ad un'immagine di partenza **x** un vettore di perturbazioni **η.**
Queste perturbazioni dovrebbero essere impercettibili all'occhio umano ma abbastanza forti da modificare la predizione della rete, traendola quindi in inganno.

Per questo attacco si è utilizzato il Fast Gradient Signed Method (FGSM) [2] di Foolbox. FGSM rientra nella categoria di attacco white-box in quanto necessita del modello di partenza per costruire degli esempi avversari. Tecniche più avanzate, non discusse in questo lavoro, sono quelle black-box, che non richiedono particolari informazioni sul modello da attaccare.

L'idea alla base di FGSM è quella di massimizzare la loss function del modello in esame mantenendo piccolo η allo stesso tempo per non rendere visibile la perturbazione.
Se J(θ, x, y) è la loss function del modello e $\nabla_x J(\theta, x, y)$ ne è il suo gradiente, allora i valori di η vengono scelti in base al prodotto tra il segno del gradiente della loss function e una ε molto piccola: $\eta = \varepsilon * \text{sign}(\nabla_x J(\theta, x, y))$.

Variando ε posso scegliere la quantità di perturbazione che aggiungo all'immagine e quindi di quanto far crescere l'errore di predizione a discapito della somiglianza con l'immagine di partenza.

Nella Figura 8 si può osservare il risultato dell'attacco su un batch di immagini scelto casualmente dal test set con ε = 0,01 in termini di accuratezza prima e dopo l'attacco.


```
Accuratezza sul batch prima dell'attacco: 0.984375
Accuratezza sul batch dopo l'attacco: 0.0
MSE tra le immagini del batch prima e dopo l'attacco: 9.844676969805732e-05
```

Figura 8: *Si osservi come si riesce a portare l'accuratezza della rete dal 98% a zero introducendo una distorsione trascurabile nell'immagine.*

## Riferimenti bibliografici

[1] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi and Siwei Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics"
[2] I.J. Goodfellow and J. Shlens and C. Szegedy, "Explaining and Harnessing Adversarial Examples," International Conference on Learning Representations (ICLR), 2015