# Hafta 12 - Çekişmeli Üretici Ağlar

**BGM 565 - Siber Güvenlik için Makine Öğrenme Yöntemleri**

Bilgi Güvenliği Mühendisliği
Yüksek Lisans Programı

**Dr. Ferhat Özgür Çatak**
ozgur.catak@tubitak.gov.tr

İstanbul Şehir Üniversitesi
2018 - Bahar

# İçindekiler

# İçindekiler

# Çekişmeli Üretici Ağlar
Generative Adversarial Networks

## Çekişmeli Üretici Ağlar

- İki arklı ağdan oluşan derin sinir ağ mimarileridir [1]
  - **Düşmanca (Adversarial)**
- Herhangi bir veri dağıtımını taklit etmeyi öğrenebilirler.
- GAN kendi başına benzer veriler oluşturması öğretilebilir:
  - resimler, müzik, konuşma, düzyazı
- **zero-sum game framework** şeklinde iki ağın birbirine itiraz etmesi üzerine kurulu bir mimaridir.

[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.

# Generative vs. Discriminative Algorithms I

## Discriminative Algorithms

- Girdi verilerinin sınıflandırmaya çalışırlar.

$$h : \mathcal{X} \rightarrow \mathcal{Y}$$

- Bir girdinin niteliklerine bakarak etiket veya ait olduğu kategorinin bulunmasıdır.
- Bir epostada bulunan kelimeler veya karakterler analiz edilerek iletinin **SPAM** veya **NOT SPAM** olarak etiketlenmesi.
  - Logistic Regression: $p(y = 1|\mathbf{x})$
  - "the probability that an email is spam given the words it contains."
- **Discriminative Algorithms**: *Nitelikler* $\Rightarrow$ *Etiketler*

# Generative vs. Discriminative Algorithms II

## Generative Algorithms

- Bu e-postanın spam olduğunu varsayalım, bu özellikler ne kadar olasıdır?
- Discriminative: $y$ ve $\mathbf{x}$ arasında ilişkiye odaklanır
- Generative : $\mathbf{x}$ nasıl elde edildi?
  - Bir spam mail nasıl oluşturulabilir.
  - $p(\mathbf{x}|y)$: $y$ bilindiğinde $\mathbf{x}$'in olasılığı. Bir sınıfa ait niteliklerin olasılığı
- Generative vs. Discriminative
  - Discriminative yöntemler sınıflar arasındaki sınırı öğrenir.
  - Generative yöntemler her bir sınıfın dağılımını modelemektedirler.

# GAN Modelleri Nasıl Çalışır? I

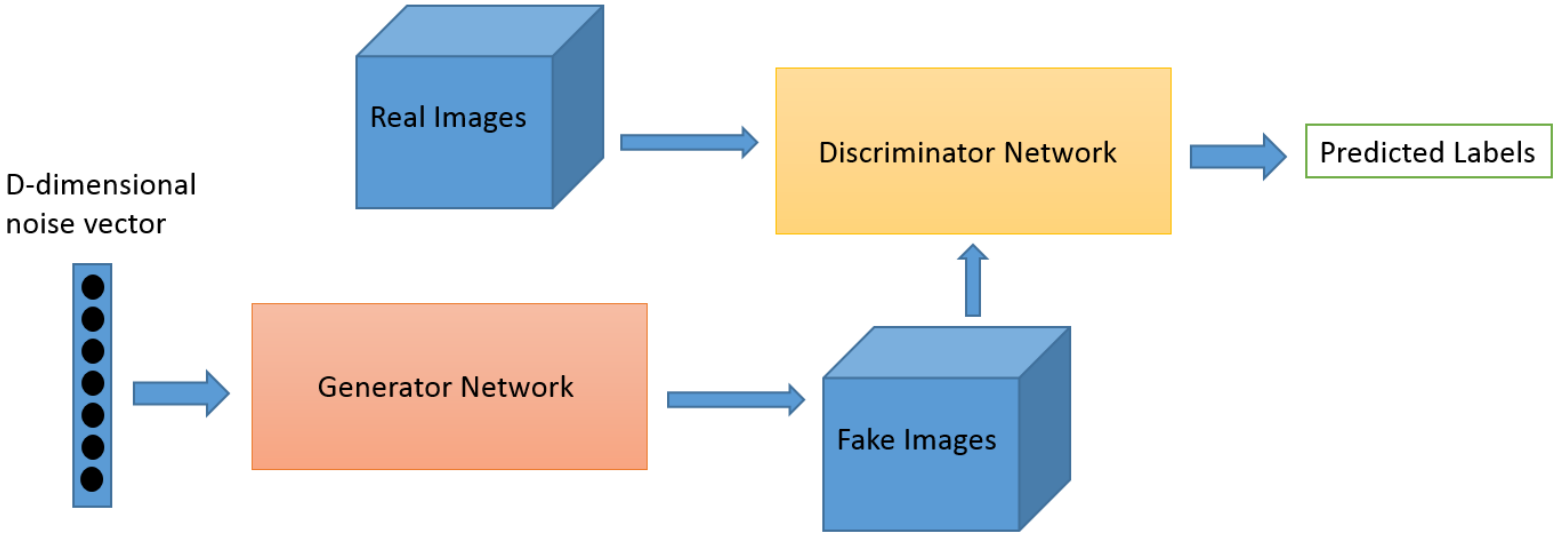
## GAN

- **Generator** olarak adlandırılan bir sinir ağı, yeni veri örneklerini üretirken, diğeri, **Discriminator**, onları doğruluk için değerlendirir.
- **Discriminator** gözden geçirdiği her veri örneğinin gerçek eğitim veri kümesine ait olup olmadığına karar verir.
- Örnek problem
  - Gerçek dünyadan alınan MNIST veri kümesinde bulunanlar gibi elle yazılmış rakamların üretimi.
  - Gerçek MNIST veri kümesinden bir örnek gösterildiğinde, Discriminator'ın amacı, onları doğru olarak tanımaktır.
  - Generator yeni örnekler oluşturup Discriminator'a iletir. Dsicriminator'ın bu örneklerin gerçek olup olmadığını anlamasını bekler.
  - Generator'ın amacı, anlaşılmadan yalan söyleyebilen, elle yazılmış rakamlar oluşturmaktır (fake images).

# GAN Modelleri Nasıl Çalışır? II

## GAN Adımları

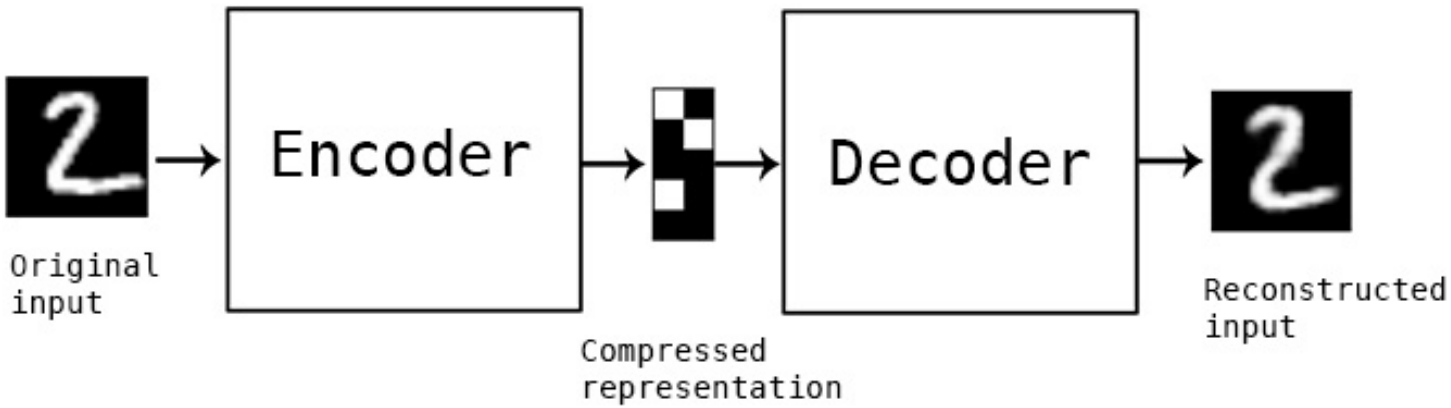
- Generator rasgele sayılar kullanarak ve bir görüntü oluşturur.
- Bu oluşturulan görüntü, gerçek veri kümesinden alınan görüntülerin akışıyla beraber Discriminator beslenir.
- Discriminator hem gerçek hem de sahte görüntüler alır ve 0 ile 1 şeklinde sınıflandırır, 1 değeri doğru, 0 ise sahte olduğunu gösterir.



# GAN vs Autoencoders

## Autoencoders

- Autoencoders giriş verilerini vektörler şeklinde olarak kodlar.
- Girdi verilerinin gizli veya sıkıştırılmış bir temsilini oluştururlar.
- GAN'lar ince, ayrıntılı ayrıntılarda veri oluştururken, AE'ler tarafından oluşturulan görüntüler daha bulanık olur.



# GAN Eğitimi

## GAN Eğitiminde İpuçları

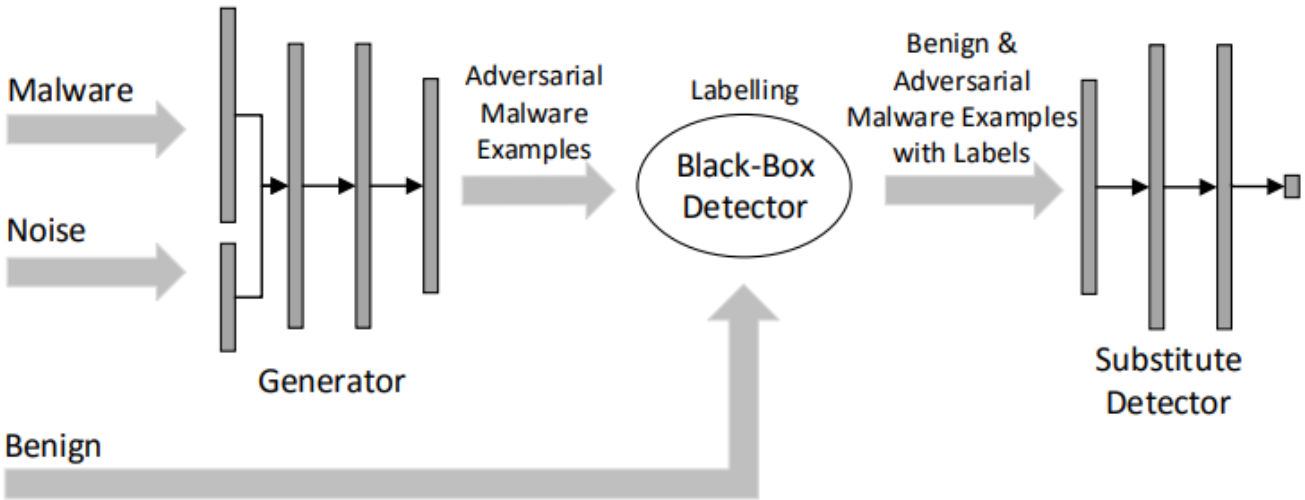
- Discriminator eğitlirken, Generator değerlerini sabit tutulur; Generator eğitilirken diskriminator Discriminator tutulur.
  - Her bir model statik bir rakibe karşı antrenman yapmalıdır.
- GAN'ların eğitilmesi uzun zaman alır. Tek bir GPU'da GAN saatlerce ve tek bir CPU'da bir günden fazla sürebilir.
- GAN unsupervised bir öğrenme çeşididir, *y* ihtiyaç yoktur .

# İçindekiler

# MalGAN I

## MalGAN [2]

- Zararlı yazılım geliştiricileri, tespit sistemlerine direk erişimleri bulunmamaktadır. **Black-Box Attacks**
- **MalGAN**: bypass black-box machine learning based detection models.
  - Generative network: minimize the generated adversarial examples' malicious probabilities
  - MalGAN is able to decrease the detection rate to nearly zero



# MalGAN II

Table 1: True positive rate (in percentage) on original samples and adversarial examples when MalGAN and the black-box detector are trained on the same training set. "Adver." represents adversarial examples.

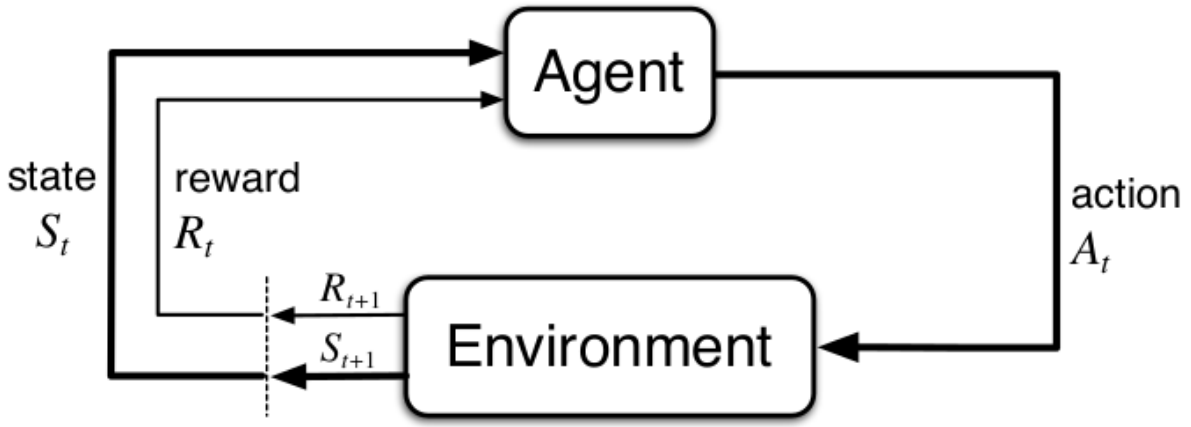
| | Training Set | | Test Set | |
|---|---|---|---|---|
| | Original | Adver. | Original | Adver. |
| RF | 97.62 | 0.20 | 95.38 | 0.19 |
| LR | 92.20 | 0.00 | 92.27 | 0.00 |
| DT | 97.89 | 0.16 | 93.98 | 0.16 |
| SVM | 93.11 | 0.00 | 93.13 | 0.00 |
| MLP | 95.11 | 0.00 | 94.89 | 0.00 |
| VOTE | 97.23 | 0.00 | 95.64 | 0.00 |

[2]Hu, W., and Tan, Y. (2017). Generating adversarial malware examples for black-box attacks based on GAN. *arXiv preprint arXiv:1702.05983*.

# gym-malware I

## OpenAI

- Kâr amacı gütmeyen yapay zeka araştırma şirketi.
- 1 milyar dolar tutarında bir bağış ile işe başlayan proje
- Destek verenler arasında Elon Musk, Sam Altman, Peter Thiel gibi isimler bulunmaktadır.
- GYM
  - Gym is a collection of environments/problems designed for testing and developing reinforcement learning algorithms



# gym-malware II

## Malware Env for OpenAI Gym

**This is a malware manipulation environment for OpenAI's** `gym`. OpenAI Gym is a toolkit for developing and comparing reinforcement learning algorithms. This makes it possible to write agents that learn to manipulate PE files (e.g., malware) to achieve some objective (e.g., bypass AV) based on a reward provided by taking specific manipulation actions.

## Objective

Create an AI that learns through reinforcement learning which functionality-preserving transformations to make on a malware sample to break through / bypass machine learning static-analysis malware detection.

**Şekil**: https://github.com/endgameinc/gym-malware

# PassGAN I

## PassGAN [3]

- It uses a GAN to autonomously learn the distribution of real passwords from actual password leaks, and to generate high-quality password guesses.

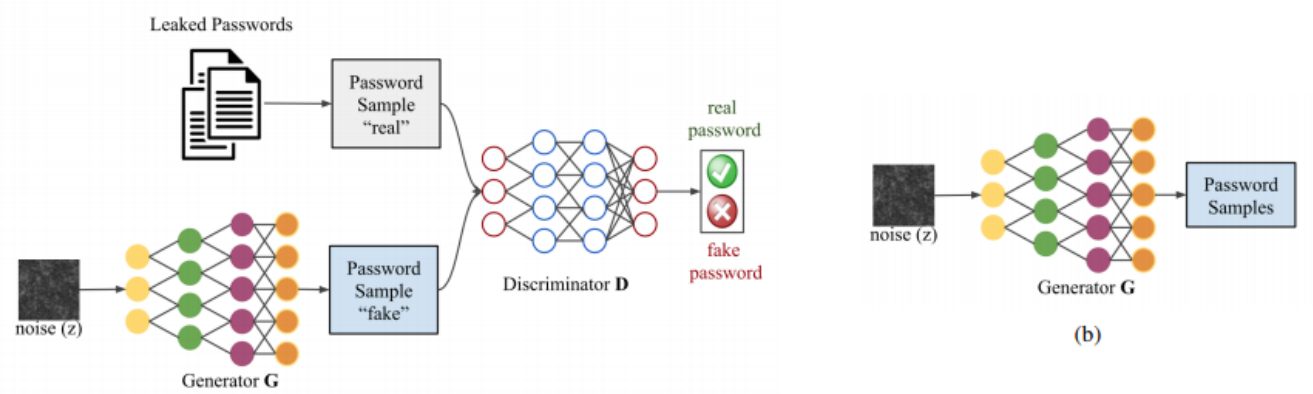



Fig. 1: Summary of PassGAN's Architecture. In the training procedure, shown in (a), the discriminator (D) processes passwords from the training dataset, as well as password samples produced by the generator (G). Based on the feedback from D, G fine-tunes its network to produce password samples that are close to the training set (G has no direct access to the training set). The password generation procedure is shown in (b).

# PassGAN II

README.md

## PassGAN

This repository contains code for the *PassGAN: A Deep Learning Approach for Password Guessing* paper.

The model from PassGAN is taken from *Improved Training of Wasserstein GANs* and it is assumed that the authors of PassGAN used the improved_wgan_training tensorflow implementation in their work. For this reason, I have modified that reference implementation in this repository to make it easy to train ( `train.py` ) and sample ( `sample.py` ) from. This repo contributes:

- A command-line interface
- A pretrained PassGAN model trained on the RockYou dataset

[3] Hitaj, Briland, et al. "Passgan: A deep learning approach for password guessing." *arXiv preprint arXiv:1709.00440 (2017)*.

# SSGAN: Secure Steganography Based on Generative Adversarial Networks I

## SSGAN [4]

- **Steganografi**: normal görünümlü dosyalarda bilgi gizleme işlemidir.
  - Changing the least significant bit in each RGB pixel value of an image would allow information to leak without ruining the image for human perception.
  - Statistically, however, these images are easy to detect.

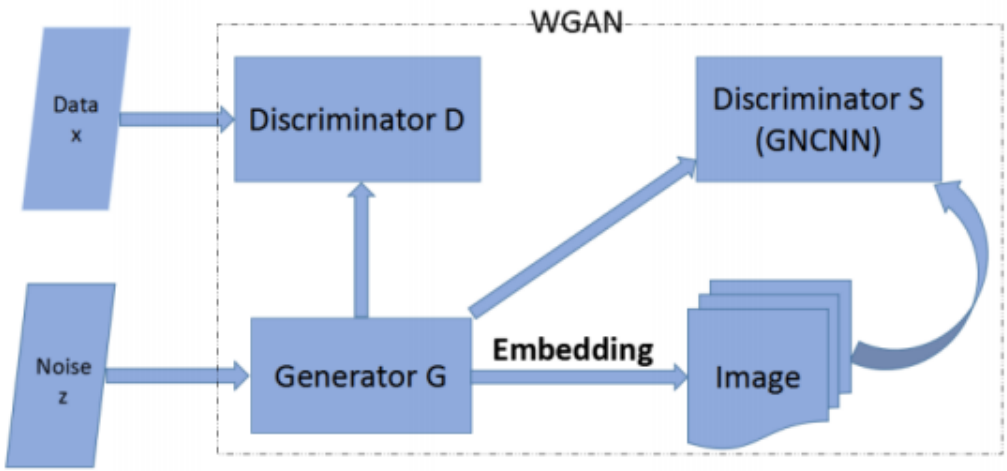



**Fig. 1.** The SSGAN model

# SSGAN: Secure Steganography Based on Generative Adversarial Networks II

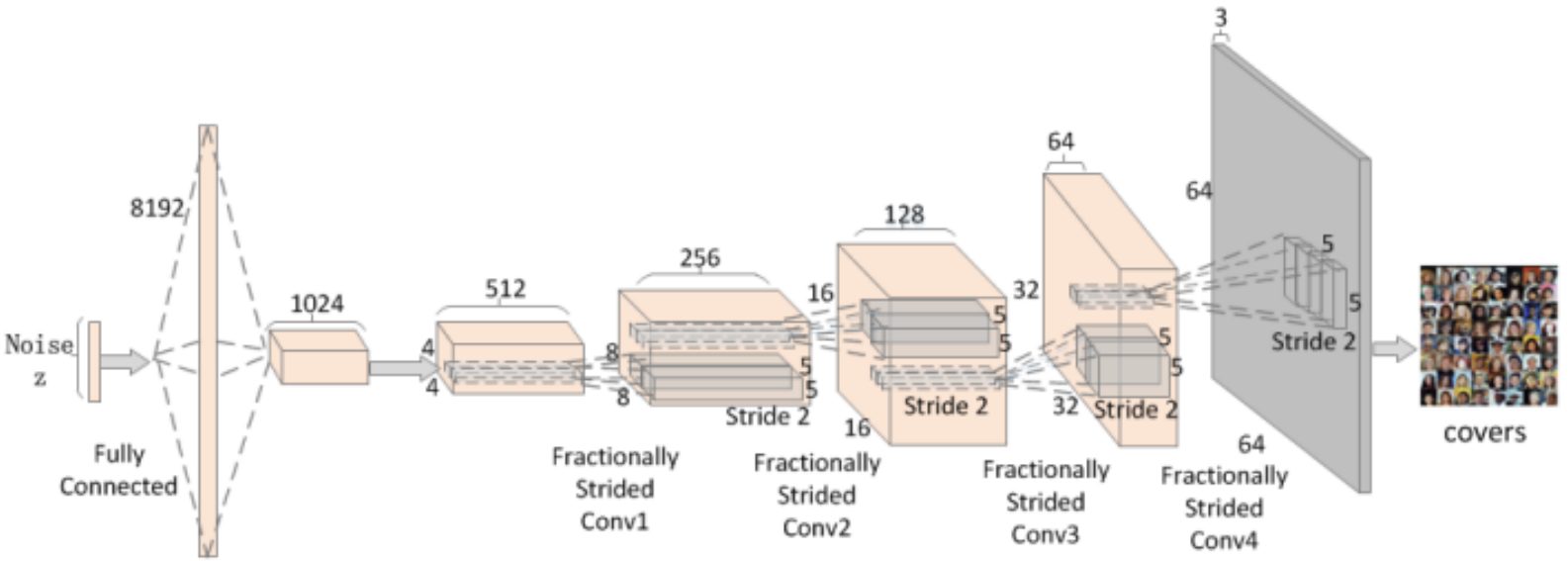



**Fig. 2.** The generative network structure

# SSGAN: Secure Steganography Based on Generative Adversarial Networks III

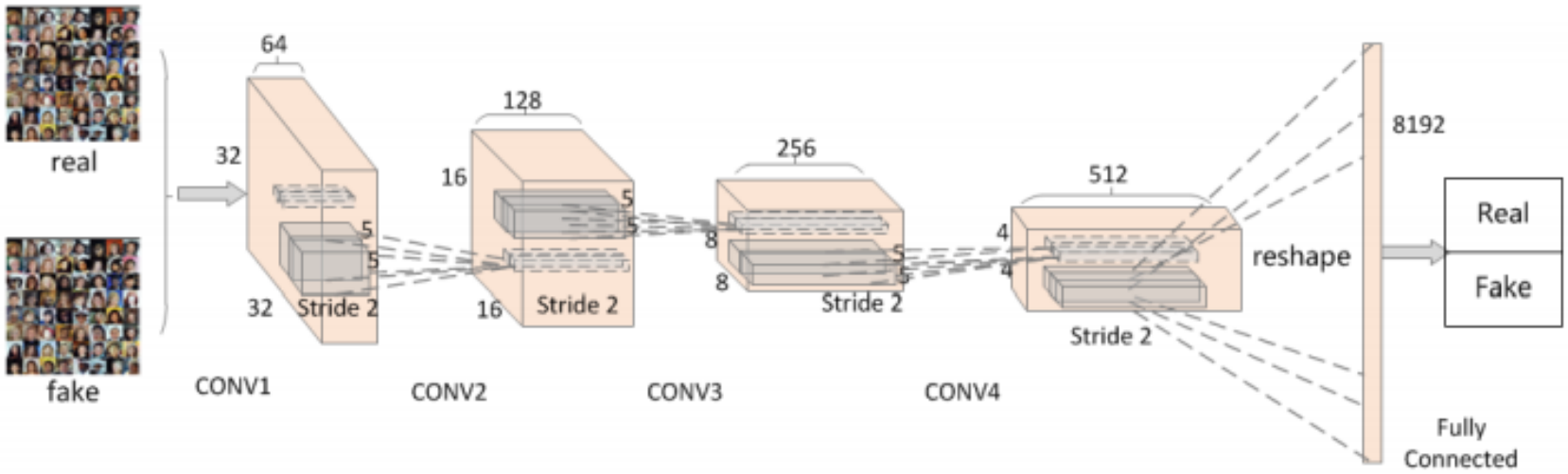



**Fig. 3.** The discriminative network structure

# SSGAN: Secure Steganography Based on Generative Adversarial Networks IV

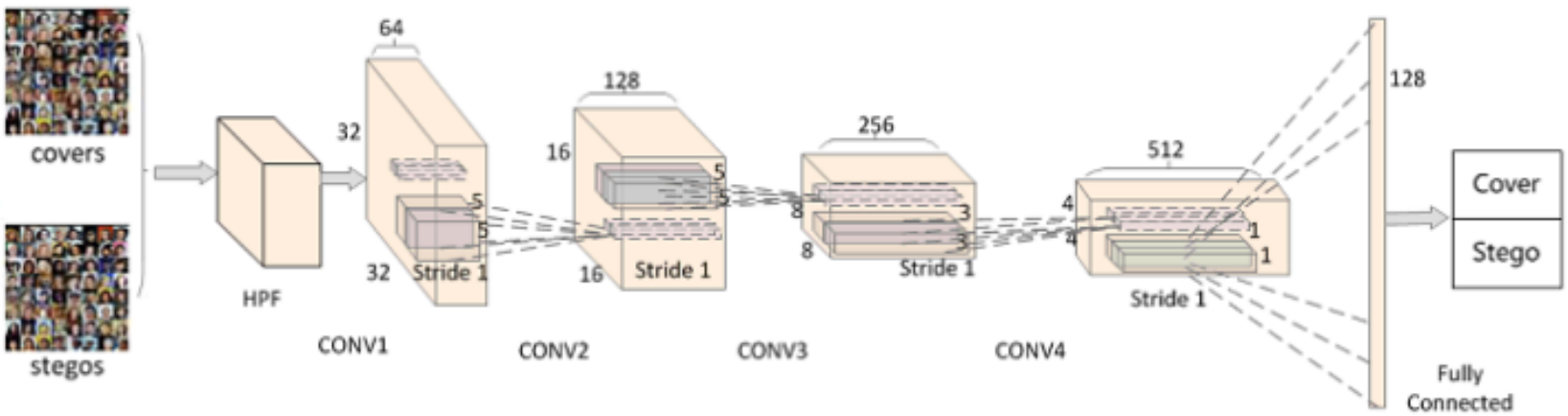



**Fig. 4.** The steganalysis network structure

[4]Shi, Haichao, et al. "Ssgan: Secure steganography based on generative adversarial networks." *arXiv preprint arXiv:1707.01613 (2017)*.