Adversarial Placement Vector Learning

Ayesha Rafique, Tauseef Iftikhar, Nazar Khan

{mscsf16m020, tauseef.iftikhar, nazarkhan}@pucit.edu.pk

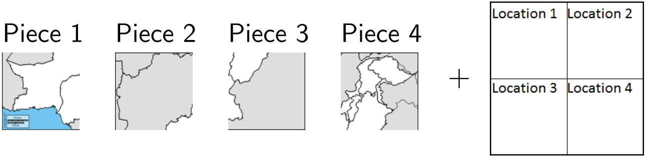
http://faculty.pucit.edu.pk/nazarkhan/

Computer Vision & Machine Learning Group
Punjab University College of Information Technology (PUCIT)
Lahore, Pakistan

ICACS 2019
Lahore, Pakistan
18th February 2019

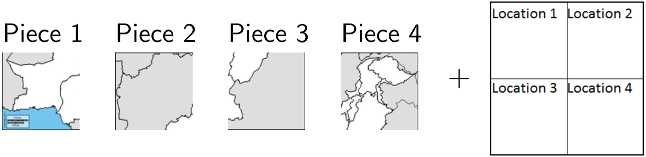
Motivation

What placement vectors do?



Motivation

What placement vectors do?



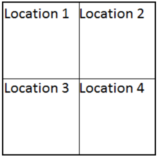
$$\begin{bmatrix} 3 \\ \\ \\ \\ \end{bmatrix}$$

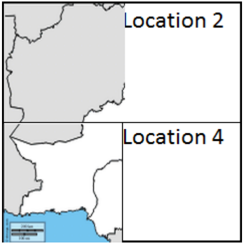
Motivation

What placement vectors do?

Piece 1 Piece 2 Piece 3 Piece 4

$+$





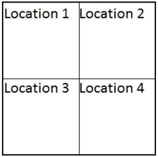
$$\begin{bmatrix} 3 \\ 1 \\ \\ \end{bmatrix}$$

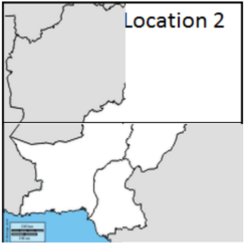
Motivation

What placement vectors do?

Piece 1 Piece 2 Piece 3 Piece 4

$+$

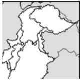




$$\begin{bmatrix} 3 \\ 1 \\ 4 \end{bmatrix}$$

Motivation

What does placement vectors do?

Piece 1	Piece 2	Piece 3	Piece 4
			

$+$

Location 1	Location 2
Location 3	Location 4

$$\begin{bmatrix} 3 \\ 1 \\ 4 \\ 2 \end{bmatrix}$$ Placement vector

Motivation

What does placement vectors do?

Piece 1 Piece 2 Piece 3 Piece 4

+

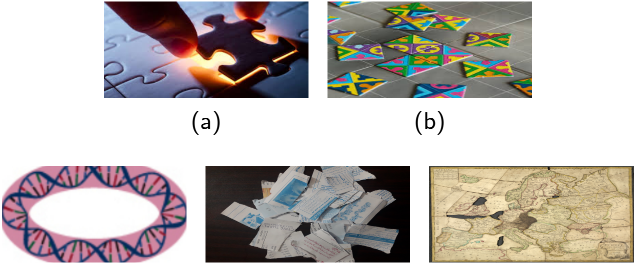
Location 1	Location 2
Location 3	Location 4

$$\begin{bmatrix} 3 \\ 1 \\ 4 \\ 2 \end{bmatrix}$$ Placement vector

Placement vector gives piece-to-location mapping.

Jigsaw puzzle and its application

Automated jigsaw puzzle solving is a challenging problem[1] with numerous scientific applications.



(a) (b)

(c) (d) (e)

Figure: (a) is traditional jigsaw puzzle, (b) is edge-matching puzzle, (c) Mitochondrial DNA, (d) Shredded document, (e)Map pieces

[1] Erik D Demaine and Martin L Demaine. "Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity". In: *Graphs and Combinatorics* 23.1 (2007), pp. 195–208.

Related work and our contribution

- Different **optimization techniques** have been applied to automatically solve jigsaw puzzles.
  - Genetic algorithms [5],
  - Probabilistic graphical models [2],
  - Quadratic programming [1] and
  - Linear programming [7].

Related work and our contribution

- Different **optimization techniques** have been applied to automatically solve jigsaw puzzles.
  - Genetic algorithms [5],
  - Probabilistic graphical models [2],
  - Quadratic programming [1] and
  - Linear programming [7].

Our contributions:

- **Learning technique** to solve jigsaw puzzles.
- Weaknesses of an end-to-end learning model for direct piece-to-map image transformations.
- Novel piece-to-location placement vector model.
- Evaluation metric for the placement vectors.

Generative Adversarial Networks (GANs)

Figure: Adversary

Generative Adversarial Networks (GANs)

Figure: GAN is a minimax game between two players G and D

Generative Adversarial Networks (GANs)

Figure: GAN is a minimax game between two players G and D

- G's aim is to generate real looking output so as to deceive D.
- D's aims to assign **fake class** to data coming from G and **real class** to data received from its training sample.

GANs framework

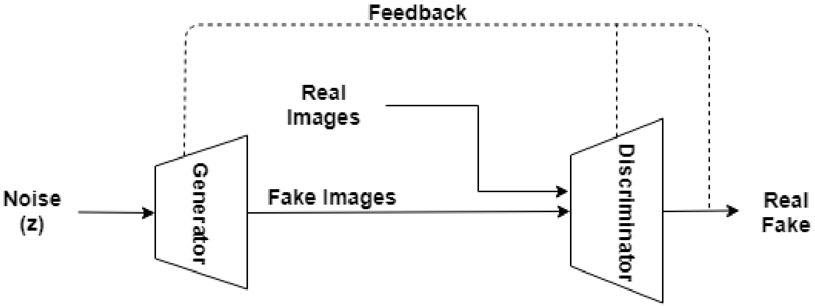


Figure: GAN training framework. The discriminator D tries to classify generated samples as fake and real samples as real. Both networks D and G are trained through feedback from D as shown by dotted arrows.

Conditional GANs (cGANs)

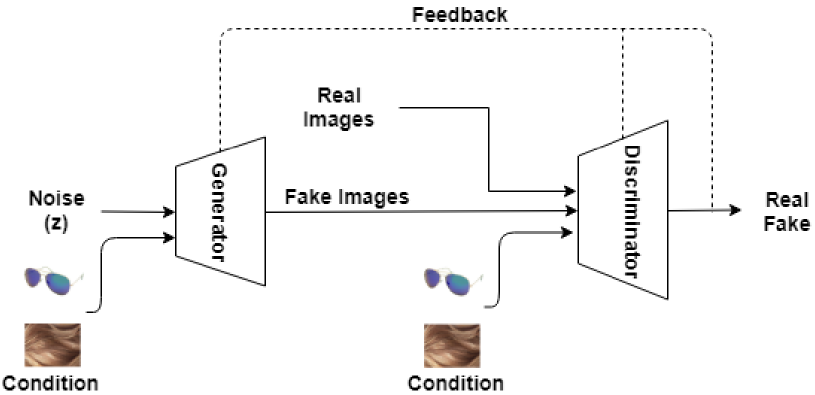


Figure: cGAN conditioned on additional input e.g. glasses and hair colour.

Our focus

Our ultimate target

- Construct a conditional model that learns piece-to-location relationship.

Our focus

Our ultimate target

- Construct a conditional model that learns piece-to-location relationship.

Our focus in the current paper

- To evaluate the effectiveness of unconditional models for learning placement vector.

Solving the jigsaw via image-to-image translation

- Jigsaw problem as an image-to-image translation task.
- Input image is random ordering of image pieces and output would be ordered image.

Figure: The jigsaw problem for square pieces – construct a meaningful image from the individual pieces.

Our first experiment

- Image-to-image translation can be effectively achieved via the Pix2Pix framework [4] which is a variation of a cGAN.
- We perform following two experiments to solve jigsaw using Pix2Pix.
    1. Experiment on shoes dataset [6].
    2. Experiment on scenes dataset [8].

Experiment on shoes dataset

Figure: Shoe images generated from shuffled input patches. While results seem satisfactory, the input patches are not reconstructed exactly.

Experiment on scenes dataset

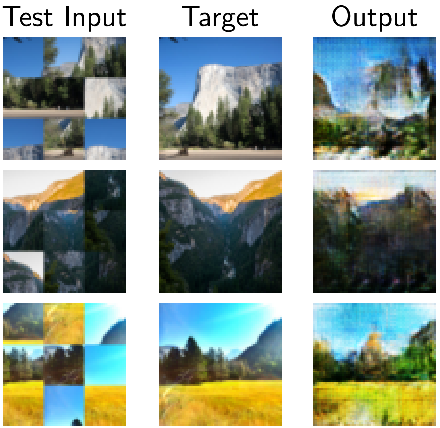


Figure: Outdoor summer scenes regenerated from shuffled input pieces. For such complex scenes, the generated images do not contain exact copies of the input pieces.

Pix2Pix analysis

Piece-to-image

- Pix2Pix does not rearrange original pieces to solve jigsaw.
- It generates each pixel of the output image.

Pix2Pix analysis

Piece-to-image

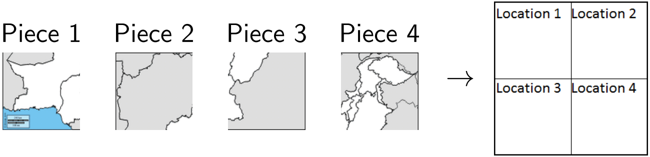
- Pix2Pix does not rearrange original pieces to solve jigsaw.
- It generates each pixel of the output image.

Piece-to-location

A novel modeling of the problem.

Placement vectors

Placement vector gives piece-to-location mapping



$$\mathbf{v} = (3, 1, 4, 2) \implies$$

Figure: Illustration of placement vectors. **Top row**: Four pieces to be placed in 2×2 grid. **Bottom row**: Placement vector $\mathbf{v}$ corresponding to correct placement of pieces.

Characteristics of placement vectors

Properties

A placement vector for P pieces must have four qualities:

1. Integer-valued entries only.
2. Range from $[1, P]$.
3. No missing numbers. Every integer from 1 to P must appear in the placement vector.
4. No duplicate numbers. An integer must appear only once.

$$\begin{bmatrix} 3 \\ 1 \\ 4 \\ 2 \end{bmatrix}$$

Training GAN for placement vectors

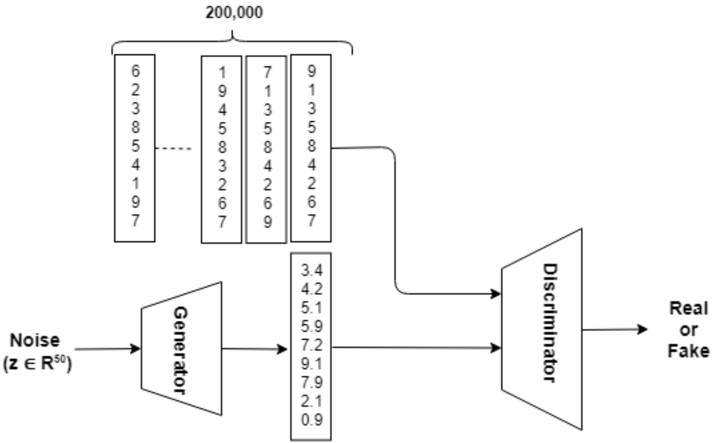


Figure: GAN was trained for 10, 000 epochs

Evaluation of placement vector

Proposed four measures to evaluate any placement vector **v**.

1. Floatingness: distance from closest integers.
   Floatingness(3.4) = abs(3.4 - 3) = 0.4
   Floatingness(3.55) = abs(3.55 - 4) = 0.45
2. Ratio of out-of-range locations
3. Ratio of missing locations
4. Ratio of duplicate locations

Sampled placement vectors

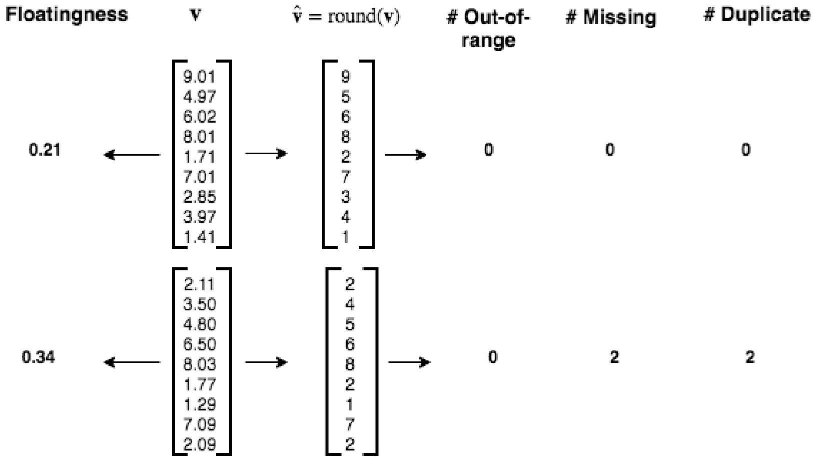


Figure: Sample placement vectors generated by our trained GAN and their evaluation by our evaluation measures.

Dealing with mode collapse

- A well-known problem with GANs is the mode collapse problem.
- Mode collapse is the situation when a GAN generates realistic but very similar samples.
- Therefore, we compute another measure of similarity among the generated placement vectors.
- We compute similarity ratio as ratio of identical vectors.

Test Results

We generate 10,000 sample placement vectors with our trained GAN and evaluate them using our five measures.

Evaluation Measures	**Average Results**
Floatingness Ratio	0.1930
Out-of-Range Ratio	0.0001
Missing Ratio	0.0533
Duplicate Ratio	0.0596
Similarity	0.0012

Table: Evaluation of placement vectors generated by our trained GAN. Range of each measure is from 0 to 1 with lower values indicating better placement vectors.

Conclusion

- Image-to-image translation models are insufficient to solve jigsaw puzzles.
- GANs successfully generate placement vectors ($z \rightarrow$ placement vector).
- We also propose five evaluation measures for evaluation of our results.

Future direction

Solving jigsaw puzzle using cGAN

References I

- Fernanda A Andalo, Gabriel Taubin, and Sione Goldenstein. "Solving image puzzles with a simple quadratic programming formulation". In: *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE. 2012, pp. 63–70.
- Taeg Sang Cho et al. "The patch transform and its applications to image editing". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.
- Erik D Demaine and Martin L Demaine. "Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity". In: *Graphs and Combinatorics* 23.1 (2007), pp. 195–208.

References II

- Phillip Isola et al. "Image-to-image translation with conditional adversarial networks". In: *arXiv preprint* (2017).
- Fubito Toyama et al. "Assembly of puzzles using a genetic algorithm". In: *Pattern Recognition, 2002. Proceedings. 16th International Conference on*. Vol. 4. IEEE. 2002, pp. 389–392.
- A. Yu and K. Grauman. "Fine-Grained Visual Comparisons with Local Learning". In: *Computer Vision and Pattern Recognition (CVPR)*. June 2014.
- Rui Yu, Chris Russell, and Lourdes Agapito. "Solving Jigsaw Puzzles with Linear Programming". In: *arXiv preprint arXiv:1511.04472* (2015).

References III

Jun-Yan Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *arXiv preprint arXiv:1703.10593* (2017).

Questions?

GANs Objective Function

Objective function of GAN is:

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim p_{data}}[\log(\underbrace{D(x)}_{\text{Discriminator output for real data x}})] + \mathbb{E}_{z \sim p_z}[\log(1 - \underbrace{D(G(z))}_{\text{Discriminator output for generated (fake) data G(z)}})] \quad (1)$$

- D wants to maximize the objective function such that $D(x)$ is close to 1 and $D(G(z))$ is close to 0.
- G wants to minimize the objective function such that $D(G(z))$ is close to 1.

cGAN Framework

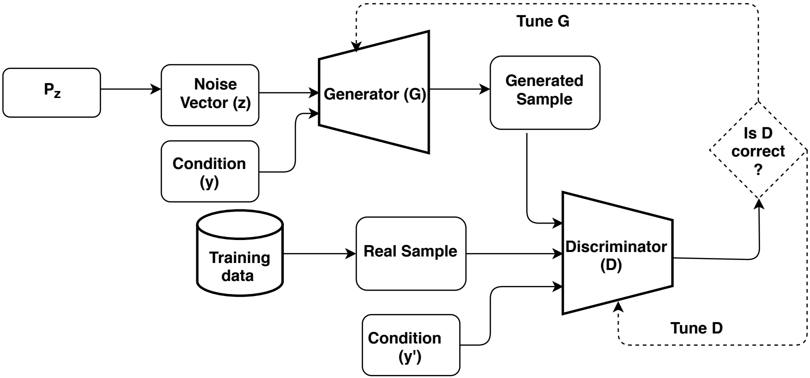


Figure: cGAN Framework.

Pix2Pix Framework

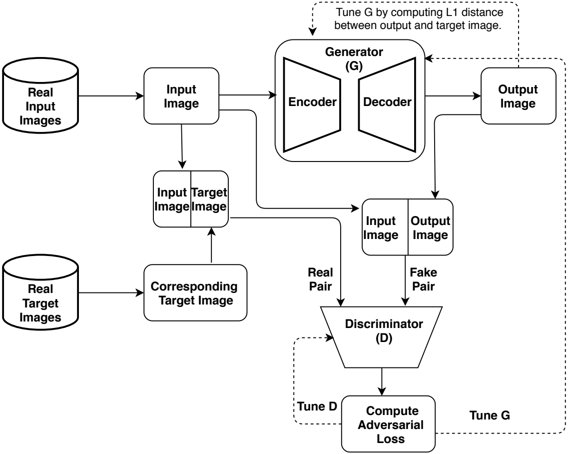


Figure: Illustration of the Pix2Pix framework.

- Mostly, Machine learning models do not produce integer-valued outputs.
- But their outputs can be evaluated for their floatingness.
- Let $\hat{\mathbf{v}} = \text{round}(\mathbf{v})$ be the closest integer-valued placement vector to $\mathbf{v}$. Then,

$$\text{Floatingness}(\mathbf{v}) = \frac{2}{P} \sum_{i=1}^{P} |v_i - \hat{v}_i| \tag{2}$$

Out-of-range ratio can be computed by counting the number of entries in $\hat{\mathbf{v}}$ that are not in the range 1 to P and dividing this count by P.

Missing ratio can be computed by counting the integers from 1 to P that do not appear in $\hat{\mathbf{v}}$ and then dividing by P.

Duplicate ratio is the number of duplicate entries in $\hat{\mathbf{v}}$ divided by $P-1$.

- Let $(\mathbf{v}_1, \ldots, \mathbf{v}_N)$ denote a set of N placement vectors generated by a GAN. Then,

$$\text{Similarity}(\mathbf{v}_1, \ldots, \mathbf{v}_N) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \mathbb{I}(L0(\mathbf{v}_i, \mathbf{v}_j), 0) \tag{3}$$

- where $\mathbb{I}(a, b) = 1$ if a and b are equal and 0 otherwise.
- And, $L0$-norm computes the Hamming distance between two vectors.