Deep Visual Analogy-Making

Yue Lu (Fudan University, Shanghai)

复旦大学

December 2015, Beijing

Outline

1. Introduction

2. Method
   - Making analogies by vector addition
   - Making analogy transformations dependent on the query context
   - Analogy-making with a disentangled feature representation

3. Experiments
   - Transforming shapes: comparison of analogy models
   - Generating 2D video game sprites
   - 3D car analogies

Introduction

Analogy

$$A \; : \; B \; :: \; C \; : \; D$$

A *is to* B as C *is to* D

Several Questions

Discriminative tasks :

- common relationship

  A ? B :: C ? D

- Are (A,B) and (C,D) related in the same way?

  A : B ? C : D

which could be formulated as classification problems.

Introduction

In this paper, they develop a novel deep network trained to perform visual analogy making.

Analogy

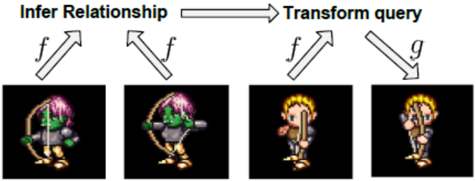
Apply the transformation $A : B$ to C

A : B :: C : ?

Visual Analogy-Making

1. recognize a visual relationship
2. generate a transformed query image

Introduction



Approach:

- Learn an encoder function

$$f : \mathbb{R}^D \ (image\ space) \rightarrow \mathbb{R}^K \ (embedding\ space)$$

- and a deep decoder function

$$g : \mathbb{R}^K \ (embedding\ space) \rightarrow \mathbb{R}^D \ (image\ space)$$

Outline

1 Introduction

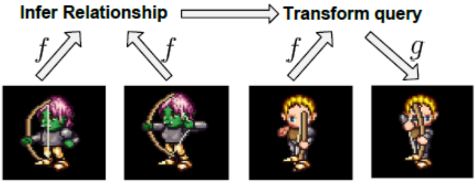
2 Method
- Making analogies by vector addition
- Making analogy transformations dependent on the query context
- Analogy-making with a disentangled feature representation

3 Experiments
- Transforming shapes: comparison of analogy models
- Generating 2D video game sprites
- 3D car analogies

Method

Input image space : $\mathbb{R}^D$

$$a \text{ is to } b \text{ as } c \text{ is to } d, \text{ where } a,\ b,\ c,\ d \in \mathbb{R}^D$$



- Learn an encoder function

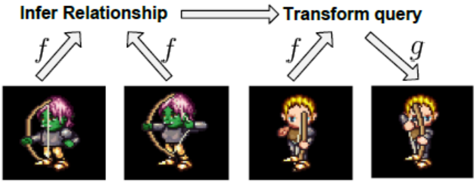
$$f : \mathbb{R}^D\ (image\ space) \rightarrow \mathbb{R}^K\ (embedding\ space)$$

- and a deep decoder function

$$g : \mathbb{R}^K\ (embedding\ space) \rightarrow \mathbb{R}^D\ (image\ space)$$

typically $K < D$.

Method



Inspiration

Some embedding methods(word2vec[21], GloVe[22])

$$d = \arg max_{w \in V} cos(f(w), f(b) - f(a) + f(c))$$

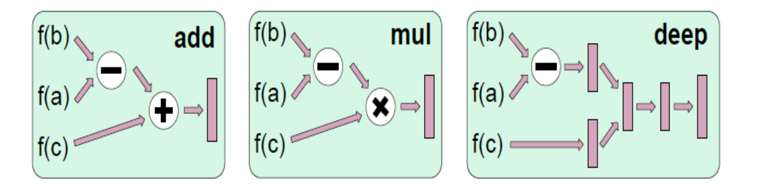
- V : vocabulary, w:word.
- $a : b :: c : d$: analogy tuple
- $f(b) - f(a) + f(c)$: vector transformation in embedding sp.

Method

Neural word representations

$$d = \arg\ max_{w \in V}\ cos(f(w), f(b) - f(a) + f(c))$$

Learn a high level representation



Making analogies by vector addition

For images (vector-addition-based analogies)

$$\mathscr{L}_{add} = \sum_{a,b,c,d \in A} ||d - g(f(b) - f(a) + f(c))||_2^2 \qquad (1)$$

it's simple to implement and train.

Two variants

1. Multiplicative interactions (between $f(b) - f(a)$ and $f(c)$)

$$\mathscr{L}_{mul} = \sum_{a,b,c,d \in A} ||d - g(f(c) + W \times_1 [f(b) - f(a)] \times_2 f(c))||_2^2 \quad (2)$$

where $W \in \mathbb{R}^{K \times K \times K}$ is a 3-way tensor.

- Define the tensor multiplication $W \times_1 v \times_2 w \in \mathbb{R}^K$ as

$$(W \times_1 v \times_2 w)_l = \sum_{i=1}^{K} \sum_{j=1}^{K} W_{ijl} v_i w_j, \quad \forall l \in \{1, \ldots, K\}.$$

where $W \in \mathbb{R}^{K \times K \times K}$ is a tensor and $v,\ w \in \mathbb{R}^K$ are vectors.

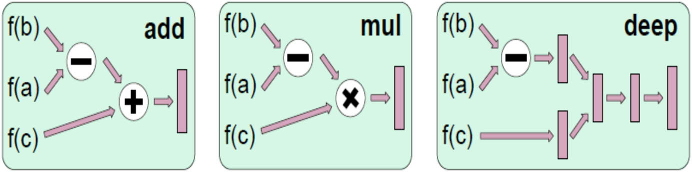
Two variants

2. Multi-layer perceptron(MLP):

$$\mathcal{L}_{deep} = \sum_{a,b,c,d \in A} ||d - g(f(c) + h([f(b) - f(a); f(c)]))||_2^2. \quad (3)$$

where $h : \mathbb{R}^{2K} \to \mathbb{R}^K$ is an MLP.

Making analogies by vector addition



$$\mathcal{L}_{add} = \sum_{a,b,c,d \in A} ||d - g(f(b) - f(a) + f(c))||_2^2 \quad (1)$$

$$\mathcal{L}_{mul} = \sum_{a,b,c,d \in A} ||d - g(f(c) + W \times_1 [f(b) - f(a)] \times_2 f(c))||_2^2 \quad (2)$$

$$\mathcal{L}_{deep} = \sum_{a,b,c,d \in A} ||d - g(f(c) + h([f(b) - f(a); f(c)]))||_2^2. \quad (3)$$

- Optimize the above objectives teaches the model to predict analogy completions in image space.

Making analogy transformations

But in order to traverse image manifolds as in Algorithm 1,

Algorithm 1: Manifold traversal by analogy, with transformation function T (Eq. 5).

Given images a, b, c, and N (# steps)
$z \leftarrow f(c)$
for $i = 1$ *to* N **do**
  $z \leftarrow z + T(f(a), f(b), z)$
  $x_i \leftarrow g(z)$
return generated images x_i $(i = 1, ..., N)$

we also want return generated images x_i $(i = 1, \ldots, N)$ accurate analogy completions in the embedding space.

Regularizer

$$R = \sum_{a,b,c,d \in A} ||f(d) - f(c) - T(f(a), f(b), f(c))||_2^2 , \quad (4)$$

makes the predicted transformation increment $T(f(a), f(b), f(c))$ match the difference of encoder embeddings $f(d) - f(c)$.

Making analogy transformations

The overall training objective

$$\mathscr{L} + \alpha R$$

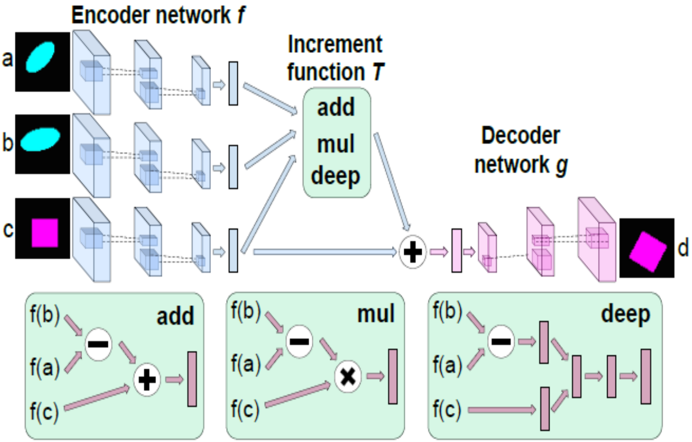
where set α by cross validation.

$$R = \sum_{a,b,c,d \in A} ||f(d) - f(c) - T(f(a), f(b), f(c))||_2^2 \,, \quad (4)$$

$$T(x,\ y,\ z) = \begin{cases} y - x & \text{when using } \mathscr{L}_{add} \\ W \times_1 [y \ - x] \times_2 z & \text{when using } \mathscr{L}_{mul} \\ MLP([y - x;\ z]) & \text{when using } \mathscr{L}_{deep} \end{cases}$$

- All parameters were trained with backpropagation using stochastic gradient descent (SGD).

Making analogy transformations



Analogy-making with a disentangling

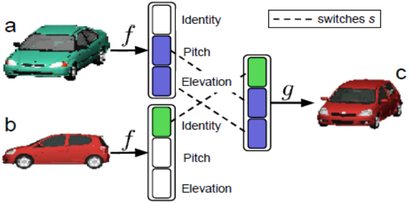
Feature disentangling

Visual analogies

- change some aspects of a query image, and leave others unchanged;
- for example, changing the viewpoint but preserving the shape and texture of an object.

To exploit this fact, they incorporate disentangling into the analogy prediction model.

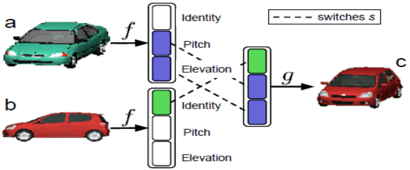
Analogy-making with a disentangling



Unlike analogy training, disentangling only requires

- 3-tuple of images a, b, c
  - a pair from which to extract hidden units, (a,b)
  - a third to act as a target for prediction, c.
- a switch unit vector s
  - s describes the sense in which a, b and c are related.

Analogy-making with a disentangling



Learning a disentangled representation

- switch unit

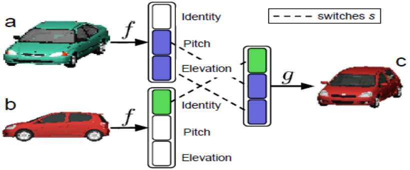
$$s \in \{0,1\}^K$$

  decides which elements from $f(a)$ and which from $f(b)$ will be used to form

$$\textit{the hidden representation} \quad z \in \mathbb{R}^K$$

- decoder $g(z) : z \rightarrow \textit{image space}$.

Analogy-making with a disentangling



The disentangling objective can be written as:

$$\mathscr{L}_{dis} = \sum_{a,b,c,s \in D} ||c - g(s \cdot f(a) + (1-s) \cdot f(b))||_2^2 \quad (6)$$

where switches s would be a block $[0;1;1]$ vector.

Analogy-making with a disentangling

Algorithm 2: Disentangling training update. The switches s determine which units from $f(a)$ and $f(b)$ are used to reconstruct image c.

Given input images a, b and target c
Given switches $s \in \{0,1\}^K$
$z \leftarrow s \cdot f(a) + (1 - s) \cdot f(b)$
$\Delta\theta \propto \partial/\partial\theta \left(||g(z) - c||_2^2\right)$

Algorithm 2 describes the learning update we used to learn a disentangled representation.

Outline

1. Introduction

2. Method
   - Making analogies by vector addition
   - Making analogy transformations dependent on the query context
   - Analogy-making with a disentangled feature representation

3. Experiments
   - Transforming shapes: comparison of analogy models
   - Generating $2D$ video game sprites
   - $3D$ car analogies

Experiments

3 datasets.

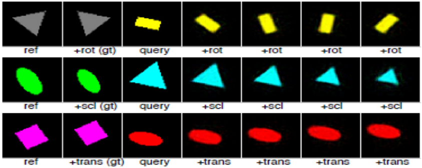
- 2D colored shapes
- 2D sprites from the Liberated Pixel Cup
- 3D car model renderings

Transforming shapes

Compare performance trained with $\mathscr{L}_{add}$, $\mathscr{L}_{mul}$ and $\mathscr{L}_{deep}$ respectively.

- rotation
- scaling
- translation

Figure 4: Analogy predictions made by $\mathscr{L}_{deep}$ for rotation, scaling and translation, respectively by row.



$\mathscr{L}_{add}$ and $\mathscr{L}_{mul}$ perform as well for scaling and transformation, but fail for rotation.

Transforming shapes

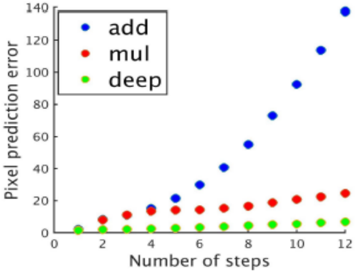
Table 1. shows that $\mathscr{L}_{add}$ and $\mathscr{L}_{mul}$ perform similarly for scaling and translation, but only $\mathscr{L}_{deep}$ can perform accurate rotation analogies.

Model	Rotation steps				Scaling steps				Translation steps			
	1	2	3	4	1	2	3	4	1	2	3	4
$\mathcal{L}_{add}$	8.39	11.0	15.1	21.5	5.57	6.09	7.22	14.6	5.44	5.66	6.25	7.45
$\mathcal{L}_{mul}$	8.04	11.2	13.5	14.2	4.36	4.70	5.78	14.8	4.24	4.45	5.24	6.90
$\mathcal{L}_{deep}$	**1.98**	**2.19**	**2.45**	**2.87**	**3.97**	**3.94**	**4.37**	**11.9**	**3.84**	**3.81**	**3.96**	**4.61**

Table 1: Comparison of squared pixel prediction error of $\mathcal{L}_{add}$, $\mathcal{L}_{mul}$ and $\mathcal{L}_{deep}$ on shape analogies.

Transforming shapes

Figure 5: Mean-squared prediction error on repeated application of rotation analogies.



(suspect that) $\mathscr{L}_{deep}$ has much better performance.

Generating 2D video game sprites

How animations can be transferred to new characters by analogy?

Dataset:

- 672 total unique characters:

  500 training, 72 validation and 100 for testing.
- 7 attributes

  body, sex, hair, armor,arm, greaves,weapon.
- 5 animations each from 4 viewpoints

  spellcast,thrust,walk,slash and shoot.

Generating 2D video game sprites

- Conduct experiments using
  - $\mathscr{L}_{add}$: $\mathscr{L}_{add}$ with disentangled features.
  - $\mathscr{L}_{dis}$: $\mathscr{L}_{deep}$ without disentangled features.
  - $\mathscr{L}_{dis+cls}$: $\mathscr{L}_{deep}$ with disentangled features.

Generating 2D video game sprites

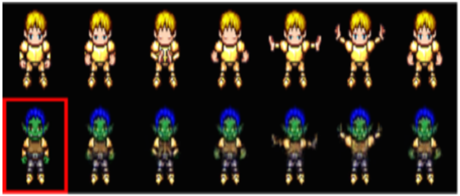


Figure 6: Transferring animations. ($\mathscr{L}_{add}$)

- the top row shows the reference.
- the bottom row shows the transferred animation.
  - where the first frame (in red) is the starting frame of a test set character.

Generating 2D video game sprites

A quantitative comparison of $\mathscr{L}_{add}, \mathscr{L}_{dis}$ and $\mathscr{L}_{dis+cls}$.

Model	spellcast	thrust	walk	slash	shoot	average
$\mathcal{L}_{add}$	41.0	53.8	55.7	52.1	77.6	56.0
$\mathcal{L}_{dis}$	40.8	55.8	52.6	53.5	79.8	56.5
$\mathcal{L}_{dis+cls}$	13.3	24.6	17.2	18.9	40.8	23.0

Table 2: Mean-squared pixel error on test analogies, by animation.

- $\mathscr{L}_{add}$ and $\mathscr{L}_{dis}$ analogy models perform similarly
- $\mathscr{L}_{dis+cls}$ wins.

Generating 2D video game sprites

Few-shot analogy-making:

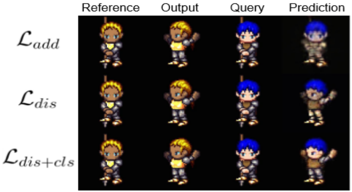


Figure 7: Few shot prediction with 48 examples.

only a small number of the target animations are provided.

Generating 2D video game sprites

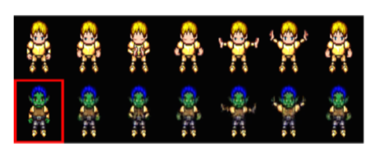
Table 3: Mean-squared pixel-prediction error for few-shot analogy transfer of the spellcast animation from each of 4 viewpoints.

	Num. of few-shot examples			
Model	6	12	24	48
$\mathcal{L}_{add}$	42.8	42.7	42.3	41.0
$\mathcal{L}_{dis}$	19.3	18.9	17.4	16.3
$\mathcal{L}_{dis+cls}$	15.0	12.0	11.3	10.4

- $\mathscr{L}_{dis}$ outperforms $\mathscr{L}_{add}$.
- $\mathscr{L}_{dis+cls}$ performs the best.

Generating 2D video game sprites

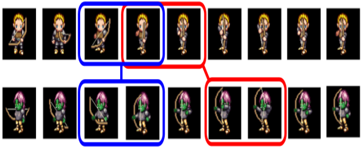
Disentangled features from 2 charactes



- Identity from Query.
- Pose from Reference.

Generating $2D$ video game sprites

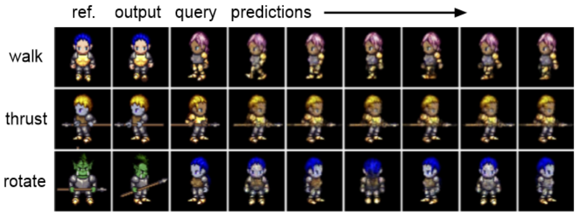
2 different characters in different viewpoints



Pose transformations were modeled by deep additive interactions used $\mathscr{L}_{dis+cls}$ to disentangle pose from identity units

Generating 2D video game sprites

Extrapolating by analogy.



The model sees the reference/output pair and repeatedly applies the inferred transformation to the query. This inference requires learning the manifold of animation poses, and cannot be done by simply combining and decoding disentangled features.

3D car analogies

- 199 car CAD models: 100 training, 49 validation and 50 testing.
- 24 rotation angles: 15 degrees/360.

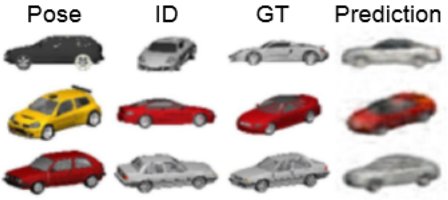


Figure 10 shows test set predictions trained on $\mathscr{L}_{dis}$.

- 4th prediction column, combine

  pose units from 1st column + identity units from the 2nd.
- GT denotes ground truth.

3D car analogies

Repeated rotation analogies in forward and reverse directions, starting from frontal pose.

which trained on $\mathscr{L}_{deep}$,

Thanks a lot for your attention.