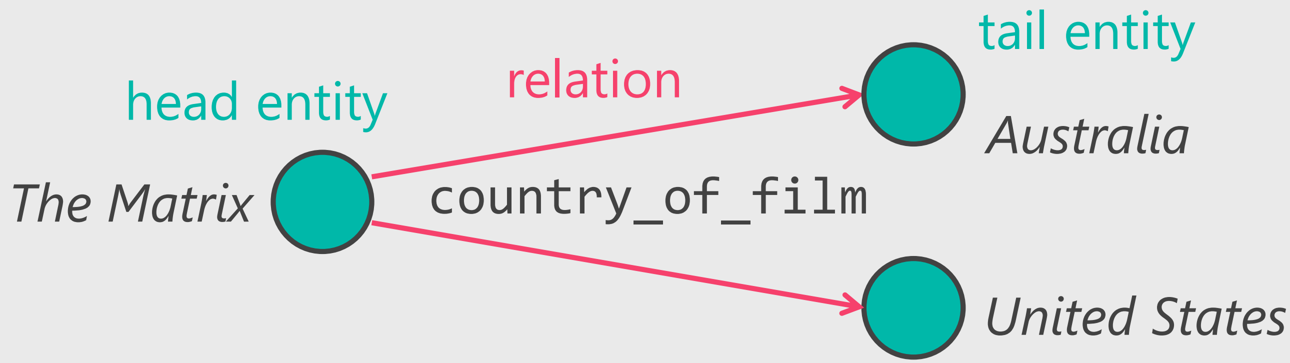
Interpretable and Compositional Relation Learning by Joint Training with an Autoencoder

Ryo Takahashi*[1] Ran Tian*[1] Kentaro Inui[1,2]
(* equal contribution)
[1]Tohoku University [2]RIKEN, Japan

Task: Knowledge Base Completion

- Knowledge Bases (KBs) store a large amount of facts in the form of <head entity, relation, tail entity> triples:



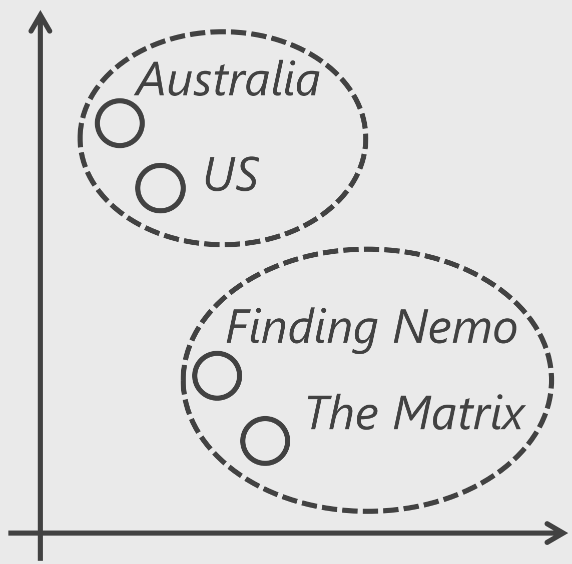
- The Knowledge Base Completion (KBC) task aims to predict missing parts of an incomplete triple:

  - Help discover missing facts in a KB

Vector Based Approach

A common approach to KBC is to model triples with a low dimension vector space, where

Entity: represented by a **low dimension vector** (so that similar entities are close to each other)



Relation: represented as **transformation** of the vector space, which can be:

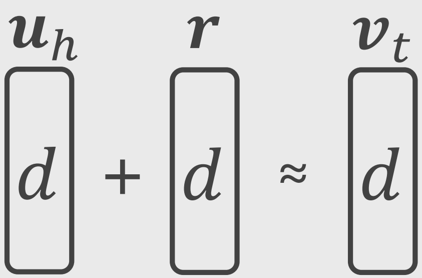
- Vector Translation
- Linear map
- Non-linear map

Up to design choice

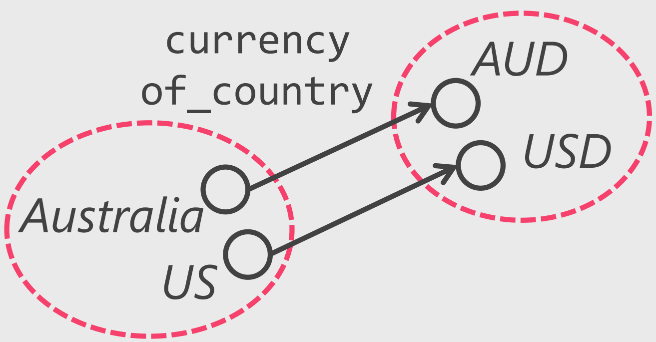
2 Popular Types of Representations for Relation

TransE [Bordes+'13]

- Relation as vector translation



- Intuitively suitable for 1-to-1 relation

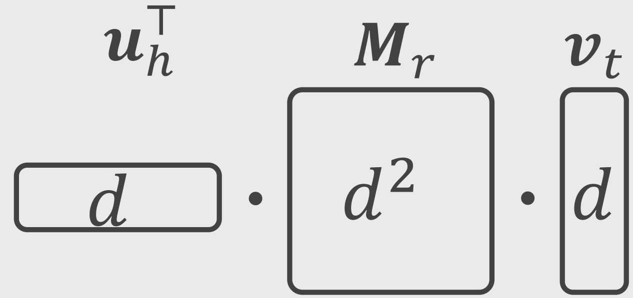


same number of entities

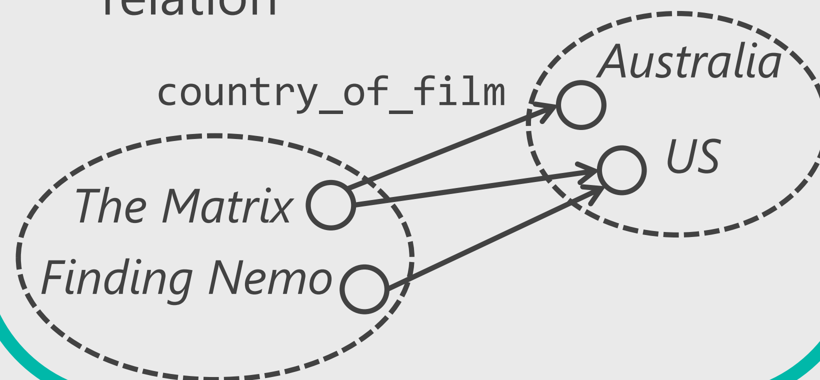
same distances within

Bilinear [Nickel+'11]

- Relation as linear transformation (matrix)



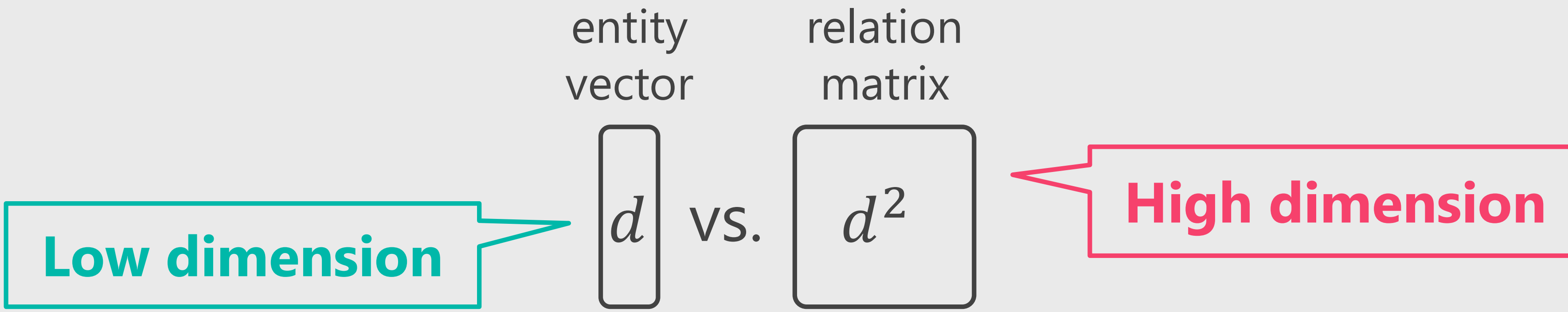
- Flexibly modeling N-to-N relation



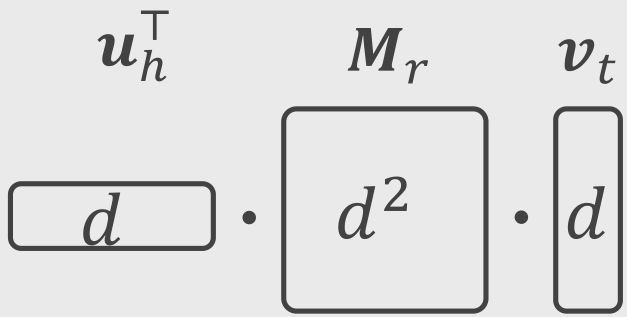
We follow

Matrices are Difficult to Train

- **More parameters** compared to entity vector



- Objective is **highly non-convex**



In this work:

① Propose jointly training relation matrices with an autoencoder:

- In order to reduce the high dimensionality

② Modified SGD with separated learning rates:

- In order to handle the highly non-convex training objective

③ Use modified SGD to enhance joint training with autoencoder

④ Other techniques for training relation matrices

Achieve SOTA on standard KBC datasets

TRAINING TECHNIQUES

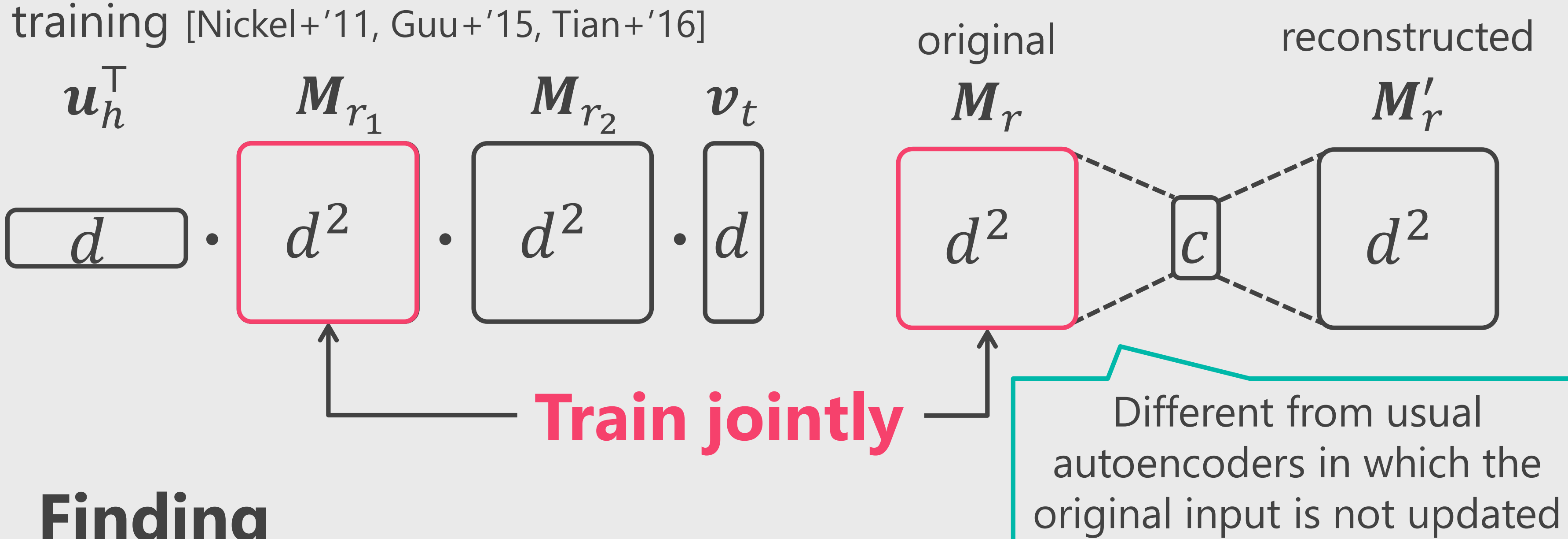
① Joint Training with an Autoencoder

Base Model

Represent relations as matrices in a **bilinear model**, can be extended with compositional training [Nickel+'11, Guu+'15, Tian+'16]

Proposed

Train an **autoencoder** to reconstruct relation matrix from low dimension coding



Finding

1. Reduce the high dimensionality of relation matrices
2. Help learn composition of relations

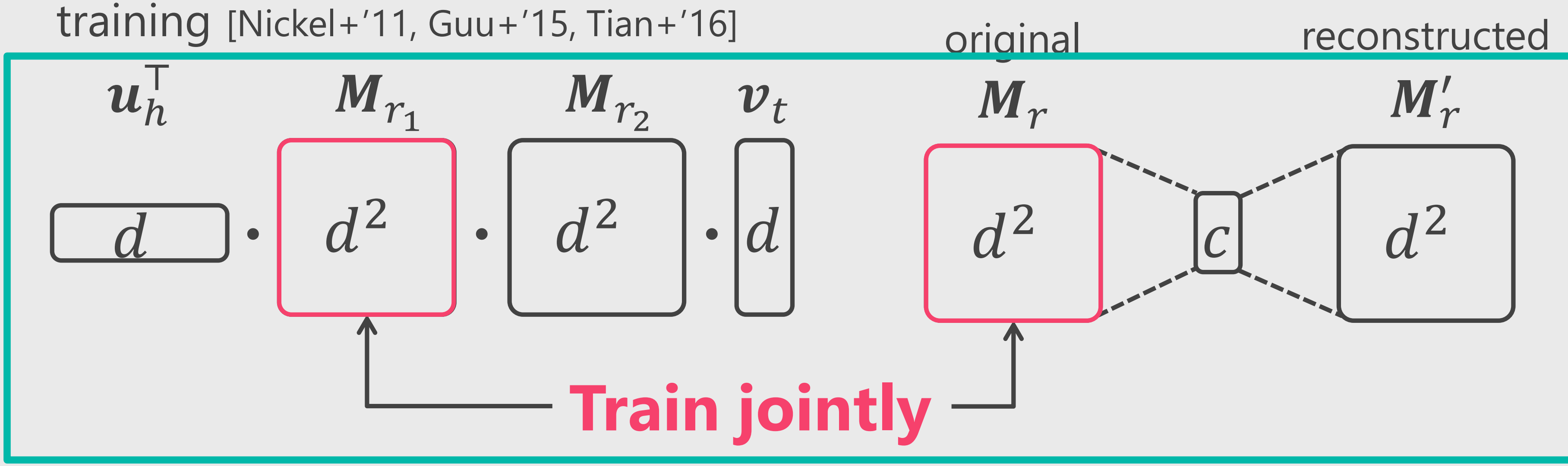
① Joint Training with an Autoencoder

Base Model

Represent relations as matrices in a **bilinear model**, can be extended with compositional training [Nickel+'11, Guu+'15, Tian+'16]

Proposed

Train an **autoencoder** to reconstruct relation matrix from low dimension coding



Not easy to carry out

Training objective is highly non-convex

→ Easily fall into local minimums

② Modified SGD (Separated Learning Rates)

Our strategy
Different learning rates for different parts of our model

Previous

The common practice for setting learning rates of SGD [Bottou, 2012]:

$$\alpha(\tau) \coloneqq \frac{\eta}{1 + \eta\lambda\tau}$$

Modified

Different parts in a neural network may have different learning rates

$$\alpha_{\mathrm{KB}}(\tau_r) \coloneqq \frac{\eta_{\mathrm{KB}}}{1 + \eta_{\mathrm{KB}}\lambda_{\mathrm{KB}}\tau_r}$$

$$\alpha_{\mathrm{AE}}(\tau_r) \coloneqq \frac{\eta_{\mathrm{AE}}}{1 + \eta_{\mathrm{AE}}\lambda_{\mathrm{AE}}\tau_r}$$

- η: initial learning rate → η_{KB}: η for KB-learning objective; η_{AE}: η for autoencoder objective
- λ: coefficient of L2-regularizer → λ_{KB}: λ for KB-learning objective; λ_{AE}: λ for autoencoder objective
- τ: counter of trained examples → τ_e: counter of each entity e; τ_r: counter of each relation r

② Modified SGD (Separated Learning Rates)

Our strategy
Different learning rates for different parts of our model

Previous

The common practice for setting learning rates of SGD [Bottou, 2012]:

$$\alpha(\tau) := \frac{\eta}{1 + \eta\lambda\tau}$$

Modified

Different parts in a neural network may have different learning rates

$$\alpha_{\mathrm{KB}}(\tau_r) := \frac{\eta_{\mathrm{KB}}}{1 + \eta_{\mathrm{KB}}\lambda_{\mathrm{KB}}\tau_r}$$

$$\alpha_{\mathrm{AE}}(\tau_r) := \frac{\eta_{\mathrm{AE}}}{1 + \eta_{\mathrm{AE}}\lambda_{\mathrm{AE}}\tau_r}$$

η: initial learning rate → η_{KB}: η for KB-learning objective; η_{AE}: η for autoencoder objective

λ: coefficient of L2-regularizer → λ_{KB}: λ for KB-learning objective; λ_{AE}: λ for autoencoder objective

τ_e: counter of each entity e

τ_r: counter of each relation r

Learning rates for frequent entities and relations can decay more quickly

② Modified SGD (Separated Learning Rates)

Our strategy

Different learning rates for different parts of our model

Rationale

NN usually can be decomposed into several parts, each one is convex when other parts are fixed

↓

NN ≈ joint co-training of many simple convex models

↓

Natural to assume different learning rate for each part

Modified

Different parts in a neural network may have different learning rates

$$\alpha_{\mathrm{KB}}(\tau_r) := \frac{\eta_{\mathrm{KB}}}{1 + \eta_{\mathrm{KB}}\lambda_{\mathrm{KB}}\tau_r}$$

$$\alpha_{\mathrm{AE}}(\tau_r) := \frac{\eta_{\mathrm{AE}}}{1 + \eta_{\mathrm{AE}}\lambda_{\mathrm{AE}}\tau_r}$$

η_{KB}: η for KB-learning objective

η_{AE}: η for autoencoder objective

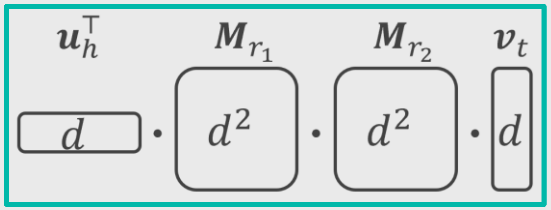
λ_{KB}: λ for KB-learning objective

λ_{AE}: λ for autoencoder objective

τ_e: counter of each entity e

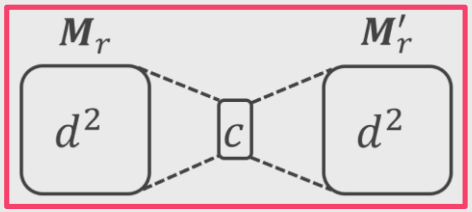
τ_r: counter of each relation r

③ Learning Rates for Joint Training Autoencoder



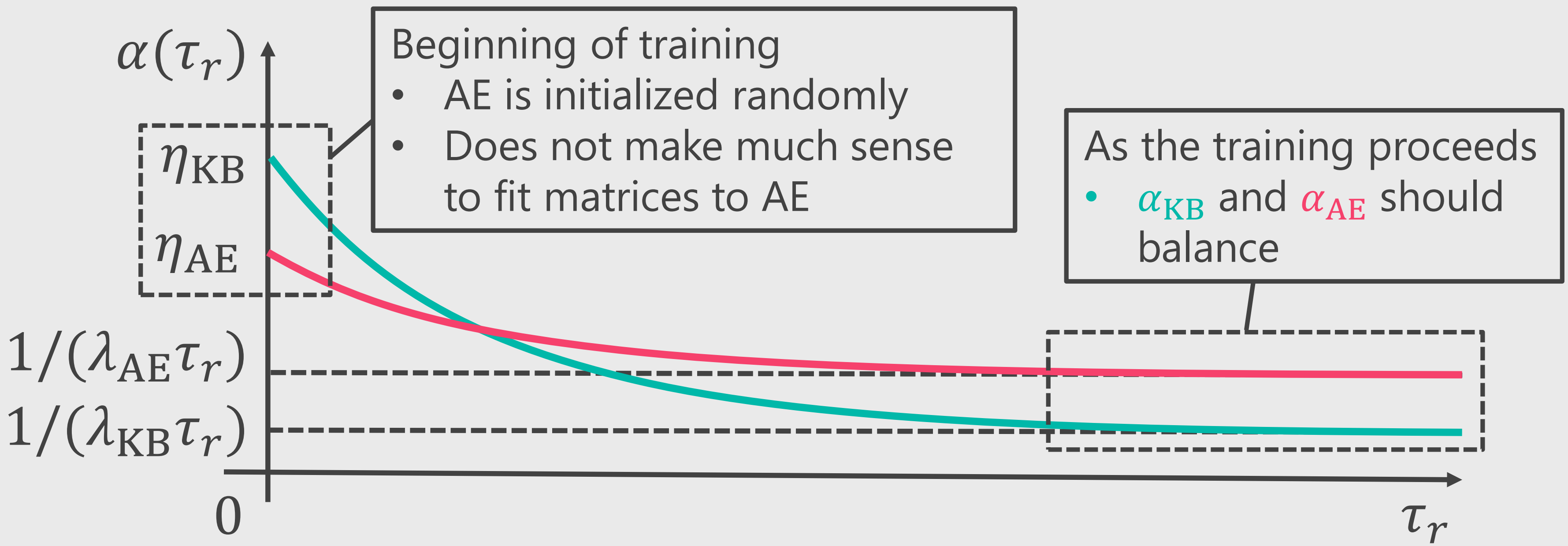
KB objective trying to predict entities

$$\alpha_{\mathrm{KB}}(\tau_r) \coloneqq \frac{\eta_{\mathrm{KB}}}{1 + \eta_{\mathrm{KB}} \lambda_{\mathrm{KB}} \tau_r}$$



Autoencoder (AE) objective trying to fit to low dimension coding

$$\alpha_{\mathrm{AE}}(\tau_r) \coloneqq \frac{\eta_{\mathrm{AE}}}{1 + \eta_{\mathrm{AE}} \lambda_{\mathrm{AE}} \tau_r}$$



④ Other Training Techniques

Normalization

normalize relation matrices to $\|\boldsymbol{M}_r\| = \sqrt{d}$ during training

$$\|\boldsymbol{M}_r\| = \sqrt{d}$$

+2.6
in Hits@10
on FB15k-237

Regularization

push $\boldsymbol{M}_r$ toward an orthogonal matrix

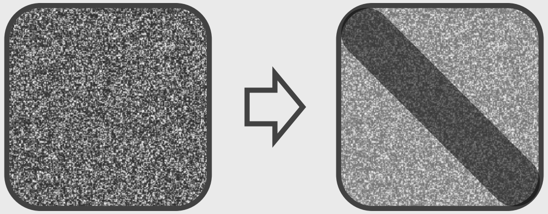
Minimize $\left\|\boldsymbol{M}_r^\top \boldsymbol{M}_r - \frac{1}{d}\mathrm{tr}(\boldsymbol{M}_r^\top \boldsymbol{M}_r) I\right\|$

+1.2
in Hits@10

Initialization

initialize $\boldsymbol{M}_r$ as $(I + G)/2$ instead of pure Gaussian

$\boldsymbol{M}_r$ $\boldsymbol{M}_r$



+0.4
in Hits@10

EXPERIMENTS

Datasets for Knowledge Base Completion

Dataset	#Entity	#Relation	#Train	#Valid	#Test
WN18RR [Dettmers+'18]	40,943	11	86,835	3,034	3,134
FB15k-237 [Toutanova&Chen'15]	14,541	237	272,115	17,535	20,466

- **WN18RR**: subset of WordNet [Miller '95]
- **FB15k-237**: subset of Freebase [Bollacker+'08]
- The previous **WN18** and **FB15k** have an information leakage issue (refer our paper for test results)
- Evaluate models by how high the model ranks the gold test triples.

Base Model vs. Joint Training with Autoencoder

Model	WN18RR			FB15k-237		
	MR ↓	MRR ↑	H10 ↑	MR ↓	MRR ↑	H10 ↑
BASE	2447	.310	54.1	203	.328	51.5
JOINT with AE	**2268**	**.343**	**54.8**	**197**	**.331**	**51.6**

Models:

- **BASE**: The bilinear model [Nickel+'11]
- **Proposed JOINT Training**: Jointly train relation matrices with an autoencoder

Metrics:

- **MR** (Mean Rank): **lower** is better
- **MRR** (Mean Reciprocal Rank): **higher** is better
- **H10** (Hits at 10): **higher** is better

Joint training with an autoencoder improves upon the base bilinear model

Compared to Previous Research

- Normalization
- Regularization
- Initialization

Model	WN18RR			FB15k-237		
	MR ↓	MRR ↑	H10 ↑	MR ↓	MRR ↑	H10 ↑
Ours						
BASE	2447	.310	54.1	203	.328	51.5
JOINT with AE	**2268**	.343	**54.8**	**197**	**.331**	**51.6**
Re-experiments						
TransE [Bordes+'13]	4311	.202	45.6	278	.236	41.6
RESCAL [Nickel+'11]	9689	.105	20.3	457	.178	31.9
HolE [Nickel+'16]	8096	.376	40.0	1172	.169	30.9
Published results						
ComplEx [Trouillon+'16]	5261	.440	51.0	339	.247	42.8
ConvE [Dettmers+'18]	5277	**.460**	48.0	246	.316	49.1

- **Base model is competitive enough**
- **Our models achieved state-of-the-art results**

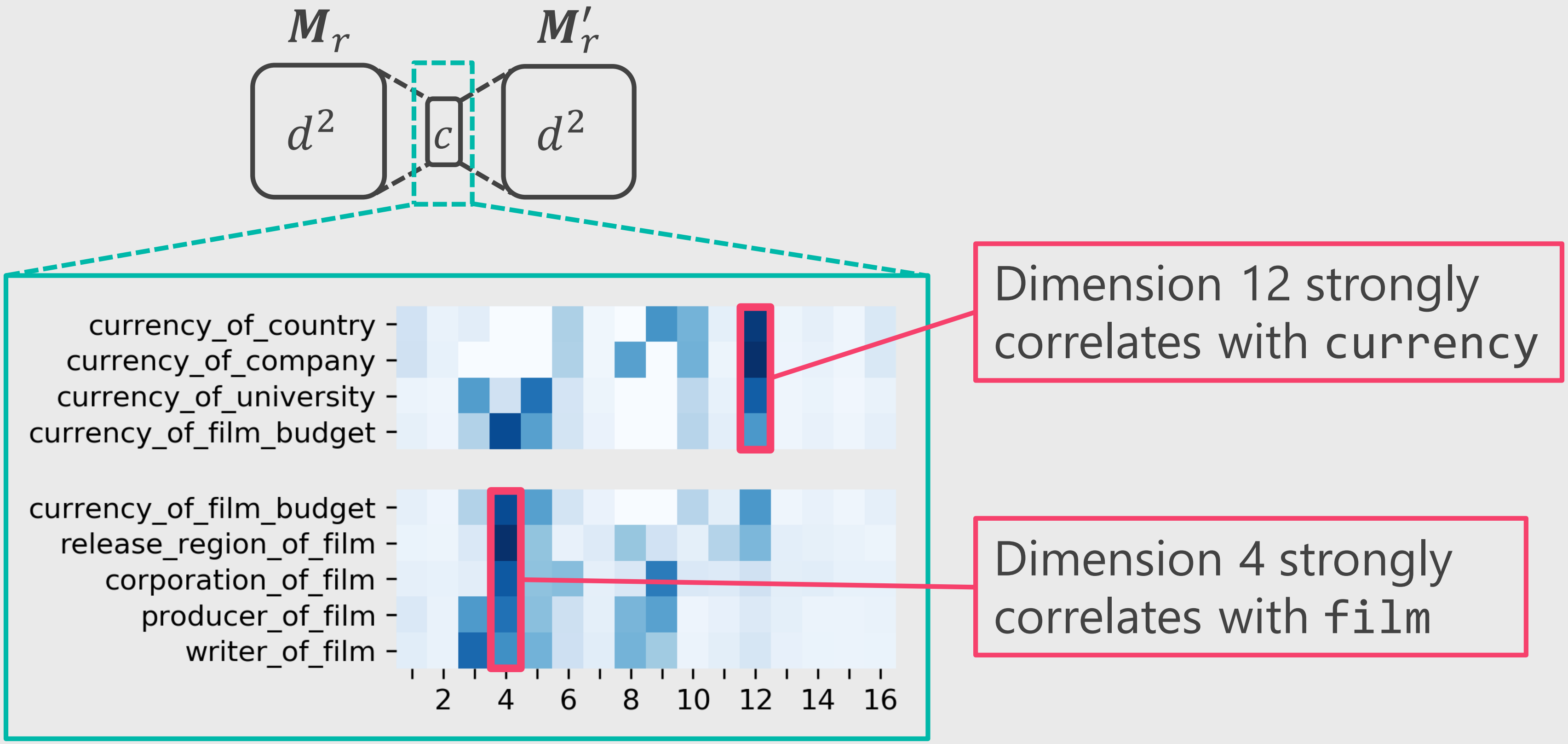
Compared to Previous Research

- Normalization
- Regularization
- Initialization

Model	WN18RR			FB15k-237		
	MR ↓	MRR ↑	H10 ↑	MR ↓	MRR ↑	H10 ↑
Ours						
BASE	2447	.310	54.1	203	.328	51.5
JOINT with AE	**2268**	.343	**54.8**	**197**	**.331**	**51.6**
Re-experiments						
TransE [Bordes+'13]	4311	.202	45.6	278	.236	41.6
RESCAL [Nickel+'11]	9689	.105	20.3	457	.178	31.9
HolE [Nickel+'16]	8096	.376	40.0	1172	.169	30.9
Published results						
ComplEx [Trouillon+'16]	5261	.440	51.0	339	.247	42.8
ConvE [Dettmers+'18]	5277	**.460**	48.0	246	.316	49.1

- **Base model is competitive enough**
- **Our models achieved state-of-the-art results**

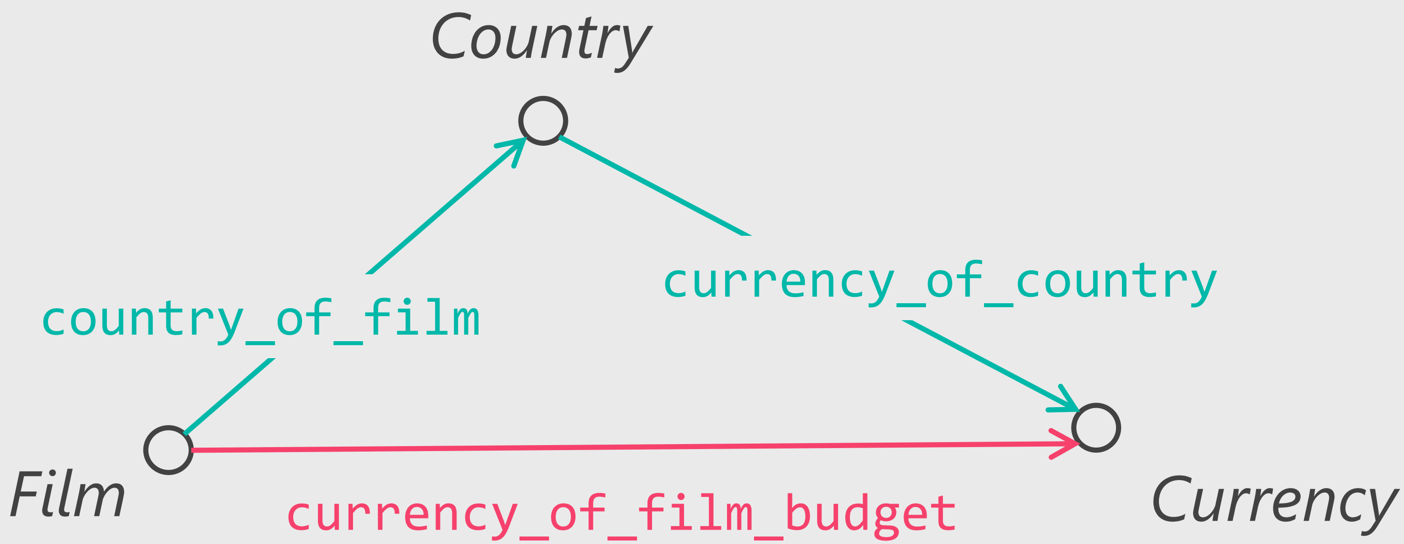
What Does the Trained Autoencoder Look Like?



- **Sparse coding of relation matrices**
- **Interpretable to some extent**

Composition of Relations

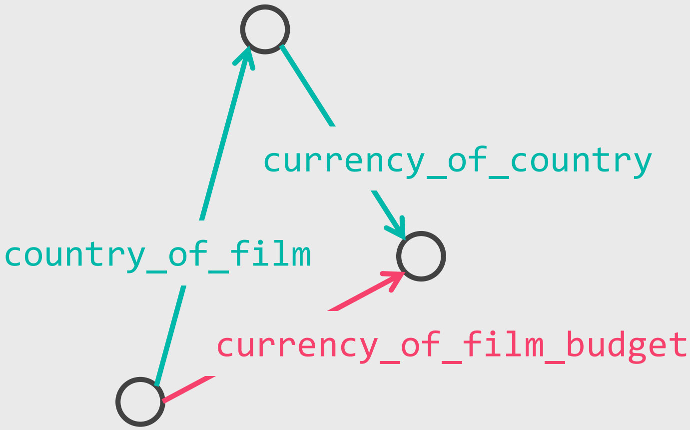
- Composition of two relations in a KB coincide with a third relation:



- Extracted 154 examples of compositional relations from FB15k-237

Joint Training Helps Find Compositional Relations

If there is a composition...



Learned relation matrices to indeed comply with the composition

$$\boldsymbol{M}_{\text{country\_of\_film}} \cdot \boldsymbol{M}_{\text{currency\_of\_country}} \approx \boldsymbol{M}_{\text{currency\_of\_film\_budget}}$$

Model	↓ MR	↑ MRR
BASE	150±3	.0280±.0010
JOINT with AE	**130±27**	**.0481±.0090**

Joint training with an autoencoder helps discovering compositional constraints

Conclusion and Discussion

Task	Knowledge Base Completion
Approach	Entities as low dimension vectors, relations as matrices
Techniques	Joint training relation matrices with autoencoder to reduce dimensionality
	Modified SGD: different learning rates for different parts
	Separated learning rates for updating relation matrices
	Normalization, Regularization, Initialization of relation matrices
Results	SOTA on WN18RR and FB15k-237
Analysis	Autoencoder learns sparse and interpretable low dimensional coding of relation matrices
	Dimension reduction helps find compositional relations
Discussion	Modern NNs have a lot of parameters
	Joint training with an autoencoder may reduce dimensionality “while the NN is functioning”
	More applications?