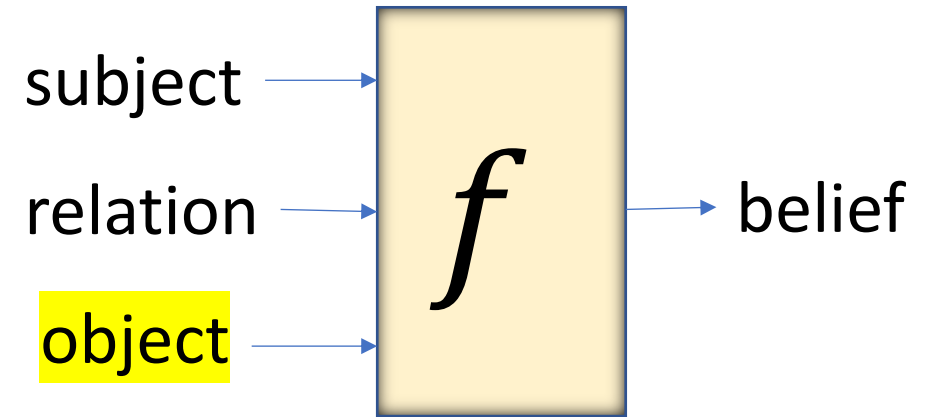


Knowledge and Retrieval

Translation and Rotation Models

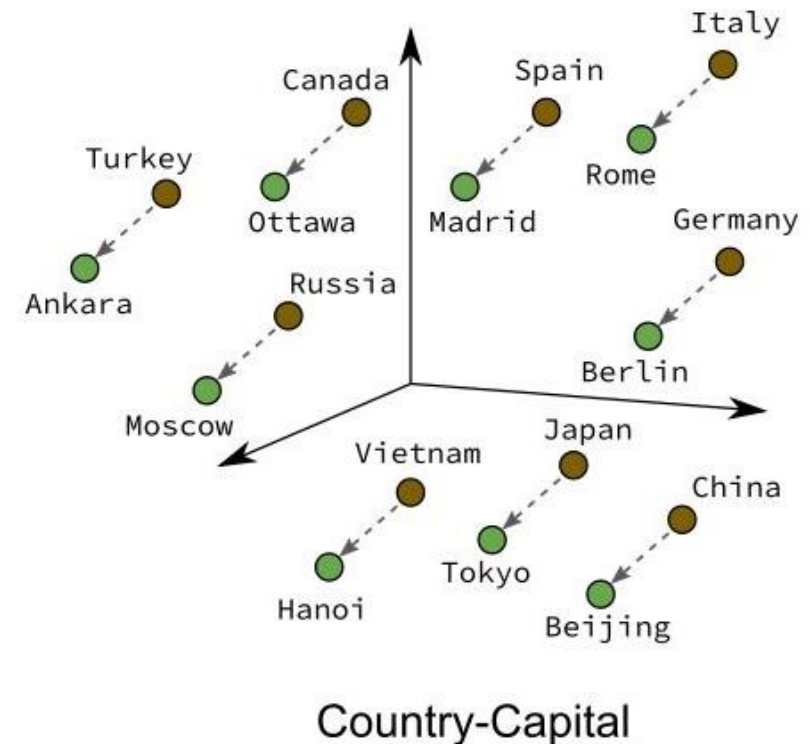
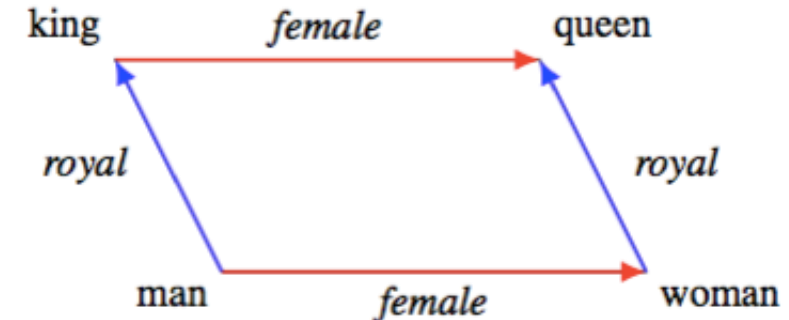
Design of scoring function f

- Design a scoring function $f(s, r, o)$ for the belief in fact $(s, r, o) \in \text{KG}$
- “In principle,” deep networks are universal function approximators
- In practice, several years and hundreds of papers proposing f and associated loss function and sampling strategy



Quick review of word2vec and GloVe

- Factor document-word matrix to obtain dense vector embeddings of words
- Related to singular value decomposition
- Each (1-1) relation can be modeled as a distinct displacement or *translation*
- \Rightarrow “TransE”



TransE

- $f(s, r, o) = \|s + r - o\|$ where $s, r, o \in \mathbb{R}^D$
- Subject, translated by relation, should be near object, otherwise low confidence that fact $(s, r, o) \in \text{KG}$
- Low score \Leftrightarrow high confidence and vice versa
- Loss adjusted to the hinge form

$$\frac{1}{K} \sum_k \max\{0, \text{margin} - f(s'_k, r, o'_k) + f(s, r, o)\}$$

Want large

Want small

TransE benefits and limitations

- 👍 Simple and fast
- 👍 Very few hyperparameters (margin and negative samples per positive sample)
- 👎 Cannot model 1-to-many, many-to-1, many-to-many relations
 - Obama + attended \approx Occidental College
 - Obama + attended \approx Harvard Law School
 - \Rightarrow Occidental College \approx Harvard Law School
- 👎 Cannot model symmetric relations
 - $(s + r = o)$ and $(o + r = s) \Rightarrow \mathbf{r = 0}$
- Many patches “XtransY”

TransH: Early fix to TransE

- Represent r by *two* artifacts
 - Hyperplane with unit normal \mathbf{p}_r
 - Displacement \mathbf{d}_r as before (was called \mathbf{r})
 - Expect $(\mathbf{s} \downarrow \mathbf{p}_r) + \mathbf{d}_r \approx (\mathbf{o} \downarrow \mathbf{p}_r)$


Subject projected
to hyperplane

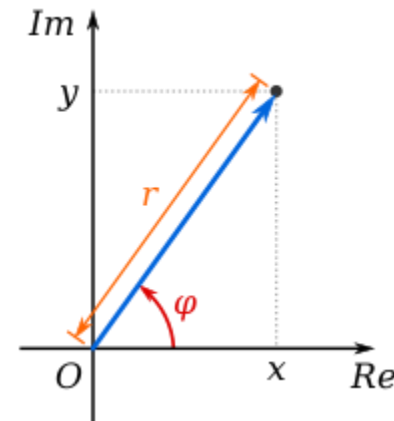
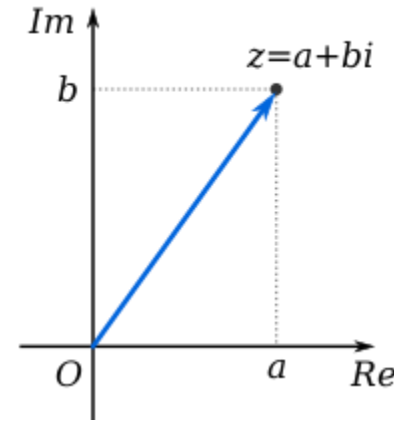
Object projected to
hyperplane

Will revisit when
we discuss
temporal
embeddings and
temporal KGs

- Many others: structured embedding (SE), FtransE, StransE, TransR, TransD, ...
- We will focus on **rotation** and **factorization**, which work better

Relation as rotation

- Complex number $a + b\sqrt{-1} = a + jb$
- Another complex number $r = \cos \theta + j \sin \theta$
- Then $(a + jb)r$ rotates (a, b) in the complex plane by angle θ anticlockwise
-  Model relation as rotation instead of translation
 - Elegant handling of anti/symmetry, inversion, composition
- Add more capacity (multiple complex dims) to handle many-to-many relations
- Close to top methods



RotatE

- Norm of complex number $a + jb$ is written as $|a + jb| = \sqrt{a^2 + b^2}$
- Let $\mathbf{s}, \mathbf{o} \in \mathbb{C}^D$ be complex vectors
- Norm of complex vector \mathbf{c} is written as $\|\mathbf{c}\| = \sqrt{\sum_d |c_d|^2} = \sqrt{\sum_d \Re(c_d)^2 + \Im(c_d)^2}$
- Let $\mathbf{r} \in \mathbb{C}^D$ with $|r_d| = 1 \ \forall d$
- $f(\mathbf{s}, \mathbf{r}, \mathbf{o}) = \|\mathbf{s} \odot \mathbf{r} - \mathbf{o}\|^2$

$$= \sum_d [\Re(s_d r_d - o_d)^2 + \Im(s_d r_d - o_d)^2]$$

Must ensure during
gradient descent

Real part

Imaginary part

KG properties supported by RotatE

- RotatE can simulate TransE
- Relation r is **symmetric** if
$$(s, r, o) \in \text{KG} \Rightarrow (o, r, s) \in \text{KG} \quad \forall s, o$$
 - $\mathbf{o} = \mathbf{s} \odot \mathbf{r}$ and $\mathbf{s} = \mathbf{o} \odot \mathbf{r} \Rightarrow \mathbf{r} \odot \mathbf{r} = \mathbf{1}$
 - I.e., rotation \mathbf{r} is its own inverse $\Rightarrow 180^\circ$ rotation
- Relation r is **anti-symmetric** if
$$(s, r, o) \in \text{KG} \Rightarrow (o, r, s) \notin \text{KG} \quad \forall s, o$$
 - For anti-symmetric relation choose a different angle

RotatE properties, continued

- Relations r and r' are **inverses** of each other if
$$(s, r, o) \in \text{KG} \Rightarrow (o, r', s) \in \text{KG} \quad \forall s, o$$
 - Inversion modeled by complex conjugate: if r is represented as $\cos \theta + j \sin \theta$, then r^{-1} is represented as $\cos \theta - j \sin \theta$
- Composition of relations is equivalent to **adding angles** of rotation
 - $r_1 \mapsto \exp(j\theta_1), \quad r_2 \mapsto \exp(j\theta_2) \Rightarrow$
$$r_1 \circ r_2 \mapsto \exp(j(\theta_1 + \theta_2))$$
 - $(e_0, r_1, e_1), (e_1, r_2, e_2)$ means $\mathbf{e}_0 \odot \mathbf{r}_1 \odot \mathbf{r}_2 = \mathbf{e}_2$