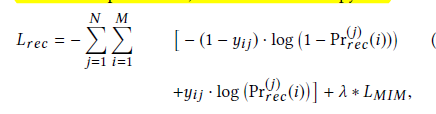
1. 使用MIM Loss预训练KG编码部分的参数
2. 用GCN对ConceptNet编码，R-GCN对DBpedia编码，最后得到word级别的wmbeddings 和item级别的embeddings 。
3. 对出现在同一个对话中的entity-word pair<e,w>，通过MIM使其表示接近。

（这一步在实际操作时，同一对话中的word都用同一个抽象表示）

1. 训练推荐模块（在这个过程中优化参数和）
2. 取得在1.中计算得出的words和embeddings的表示
3. 对于每一个对话：
   1. 表示所有出现的words的embedding matrix
   2. 表示所有出现的items的embedding matrix
   3. 使用self-attention得到一个表示，一个表示
   4. 用户表示通过在和上应用门机制得到。
   5. 计算在这一对话中每个item被推荐的概率。
4. 整个推荐过程的损失函数为（MIM损失还不懂）



其中j是会话的index, i是item的index

也即对每个对话中出现的每个item计算其推荐任务的交叉熵损失，最后再加上一个MIM损失。

1. 训练对话生成模块
2. Transformer的Encoder 部分不变。
3. 对Transformer的Decoder部分进行改进：

在原Decoder结构的两个Attention层之间加了两个KG-based Attention层

1. 在已经产生的情况下，想要计算第i个位置的输出：
   1. Decoder第n次计算最后的FFN层输出得到
   2. 第i个位置输出的概率为：



其中是使用copy机制根据两个知识图谱得出

1. 一个对话C的生成损失为：（还不懂）

