由于我的轨迹预测是为了导航服务的，所以所有的2D图像上的轨迹预测统统不考虑，因为这些轨迹预测无法提供障碍物在三维空间中的位置。

要在三维空间上预测轨迹，只用视觉是不精确的，所以我的想法是激光雷达加视觉融合。阅读了《Human Motion Trajectory Prediction: A Surve》，选择了其中采取kitti数据集的几篇论文，因为这几篇论文采取了激光雷达加视觉的方式，和我的想法较为接近，所以深入学习这几篇论文。

1. DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents

-----------------------------------------------------------------------------------------------------------

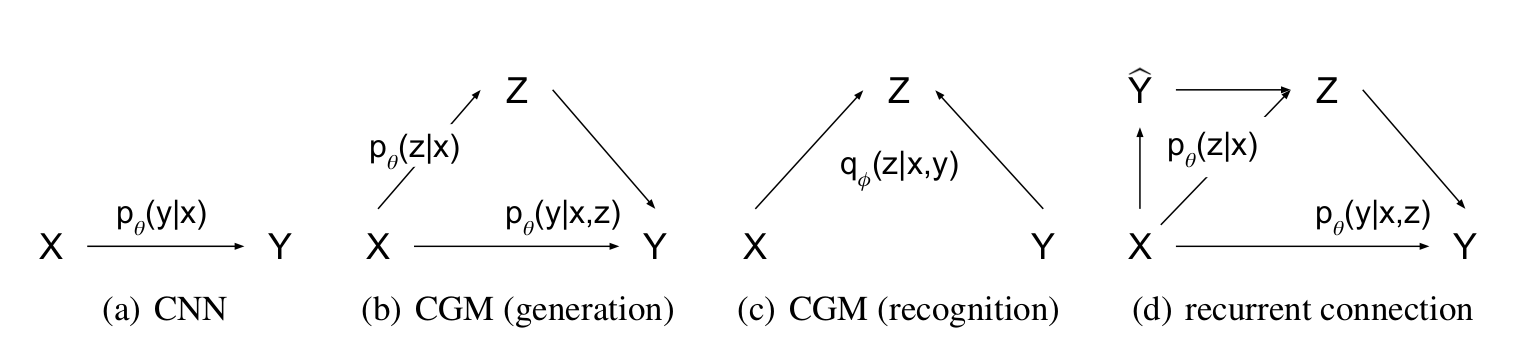
1.1辅助论文《Learning Structured Output Representation using Deep Conditional Generative Models》

这篇论文中的网络结构被DESIRE用来作对比，因此也了解一下。这篇的结构称为CVAE，是一种用来推断在状态s下出现x的概率的网络结构。这种网络通常用来预测。

CVAE是在VAE（variational auto-encoder）的基础上改进的，VAE是如下过程：

假设根据一个先验的分布概率p(z)产生了一组潜在的变量z，而在潜在变量z下，产生数据x的概率为p(x|z)。这里的理解是这样的，我们在观察中只能观察到数据x，但我们先验的知道这些数据大概和潜在的变量z有关，而且我们有一个先验的概率知道产生z的概率是多少。于是现在我们希望知道在z下产生x的概率分布。这样做有一个好处，就是x是很多的，但是z很少，从前一个状态的z过度到下一个状态的z是先验的，于是我们先得到z，在按照一定概率分布预测下一个时刻的x，这就是预测的过程。但是在整个过程中既不知道z的数量，也不知道通过z产生x的关系，甚至z的过渡过程都是不准确的，所以这个问题很困难。随机梯度变分贝叶斯（SGVB）的框架就是用来解决这个问题的。因为在这个问题中概率分布是很难学习的，所以SGVB训练了近似的分布q(z|x)。求分布q或者说p的方法是蒙特卡洛采样。这里原始的p可能是不可微的，离散的，近似的q是一个可微的函数，因此可以使用梯度下降。

整个模型的结构分为三类，如下图所示：



b的模型有一个好处，就是z->y的映射可以是一到多的。注意到b中x->y和z->y，可以近似的认为z->y。

CVAE的变化在于，认为z是由x和y产生的，即观察y和数据x双方构成了特征变量z，这里图c是一个基本结构，而d则是具体的方式，首先左上角的y是x经过一个CNN猜测的观察值，然后用这个x,y来产生z，在用z和x在真实的y中去学习分布函数。

-----------------------------------------------------------------------------------------------------------

我们来看一下《DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents》这篇论文到底是如何预测的。

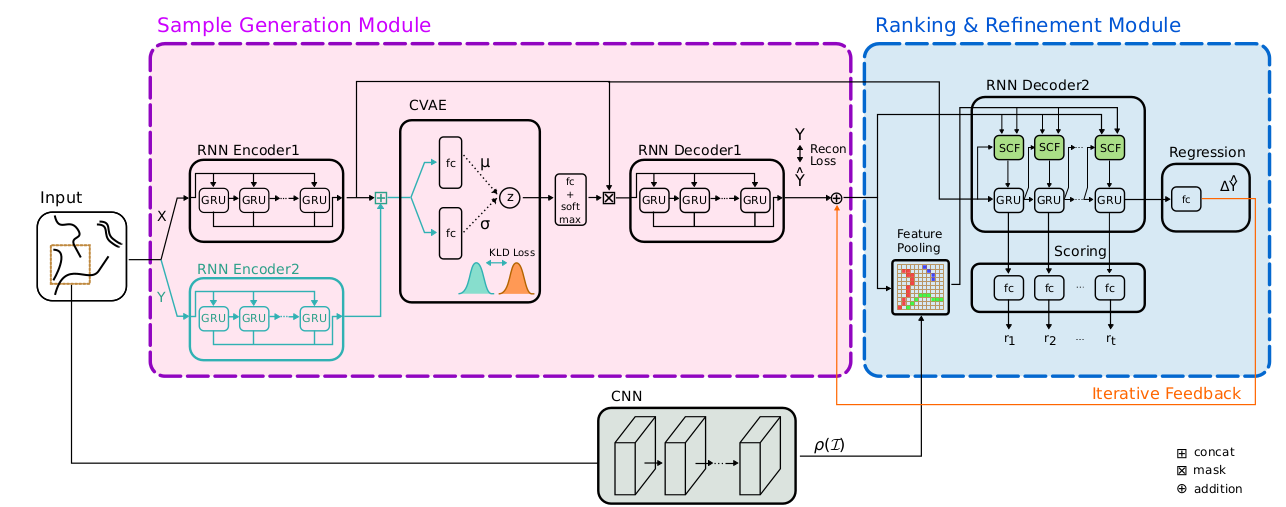
这篇文章的思路是这样的，先把图像，历史运动记录，场景的多智体的互动等信息作为输入，经过CVAE可以产生多种未来的预测轨迹。然后用一个RNN来评估这些轨迹的分数，选择最大化累计分数的预测结果。

传统的方法一般是通过历史轨迹的线性、非线性、高斯或者卡尔曼滤波等处理，得到预测的轨迹。这种做法在场景与智能体以及智能体与智能体互动较少的环境中好用。但复杂的环境下就难以预测准确了。

Inverse optimal control(IOC)或inverse reinforcement learning(IRL)构建奖励函数，可以很好的完成控制机器人或者驾驶任务，但这种方法需要先给出目标位置，即需要先提供一条可行的路线出来。并且这个方法没有考虑多智体的互动。

方法实现：

轨迹预测可以描述为如下的一个公式p(Y|X,I),其中Y是预测的轨迹(Y1,Y2,...,Yn)一共n个智能体的轨迹预测，X（X1，X2,...,Xn）一共n个智能体的历史轨迹以及I是感知信息，也就是用过去n个智能体的轨迹和感知信息来计算未来可能的n个智能体轨迹的概率。



整个网络结构如上图所示。第一部分称为Sample Generation Module，第二部分称为Ranking and Refinement Module。第一部分其实是一个CVAE结构。在训练的时候已知x对应的y，所以用两个RNN层来处理x,y，合并后通过全连接层构造潜在变量z，z在和x的特征向量经过RNN产生预测的y，训练的目标就是让预测y和真实的y尽量一致。测试时由于没有y，所以z是采样得到的，即在x下采样z，然后后面过程一样，得到预测的y。

Ranking and Refinement Module的输入是预测的多条轨迹y，通过cnn得到的图像语义信息，其他多智体的互动特征，用这些输入来评估选择哪条预测轨迹。Ranking就是对预测的多条轨迹评分，而Refine指可以从多个智能体轨迹中提取更好的预测，这种refine是考虑了所有agent的预测轨迹后的优化，即考虑了互动性以及环境因素。

网络结构的详细分析：

这个网络的输入是20帧的目标物体的x,y坐标X，训练的时候还有40帧的目标物体的x,y坐标Y，还有一张80m×80m的160×160的语义地图，

历史轨迹X是（2,20）的数组，对每一个坐标点经过一个conv1d的卷积，构成一个(16,20)的序列，卷积的kernel大小是3，padding补齐，之后经过Relu激活。这个(16,20)的数组表示一个长度20的序列，每个序列的特征维度为16。将这个序列放在GRU中，GRU中的隐状态H是一个特征维度为48的状态参数，因此经过20层GRU后输出一个48个特征维度的HX。在训练的时候，对于Y(2,40)，也使用同样的过程，得到一个同样48个特征维度的HY。

这就是图中RNN Encoder1和RNN Encoder2的部分。

接着对于CVAE部分：

首先将HX和HY合并成一个维度为96的特征向量。经过一个全连接和RELU输出(48)的特征向量。将这个特征向量同时传递给两个全连接层（图中分叉的部分），这两个全连接层都是48->48，但激活函数一个为Linear（fc1），一个为0.5exp(·)(fc2)。这里有两个48维度的向量，需要进行一个叫做reparameterization trick的操作，这个操作如下:

Z= fc1+fc2\*N(0,1),即把fc1当做是偏差量，fc2当做是方差，经过正态分布，得到特征向量Z(48维度)。

得到z后就可以对采样进行重构，预测一个轨迹Y了。

这里先经过一个fc(48->48)+softmax，得到一个48维度的概率分布。与原先RNN Encoder得到的HX进行对应元素相乘，得到一个特征的概率分布(48维度)。然后把这个特征向量放到RNN Decoder1中，这里需要特别注意，GRU的输入，只有第一个是用特征向量(48)，后面都是0，初始的H也是0，隐状态H也是48维度，因此最后输入了一个(48,40)的序列。这个序列的每一个隐状态都经过全连接输出(48->2),得到了(2,40)的序列。

需要注意的有两点，这个(2,40)的序列不是绝对位置，而是相对前一帧的偏差。另外，全连接层是一个，即一个全连接层的参数共享了这40个序列的计算。

以上这部分就得到了一个初始的预测轨迹。

Ranking 和refine的部分：

首先是图像处理，输入是语义地图(160\*160\*4),包括障碍物，绿植，车道，和行人道。经过两个Conv2d(160\*160\*4->80\*80\*16->80\*80\*32,Relu)，得到一个特征地图。

得到的这个地图需要做feature pooling，做法如下：

对于上一层得到的初始预测轨迹Yi，t，找到特征地图上对应的位置，取得值，于是得到(32,40)的特征向量。

此外，Y\_i,t和Y\_i,t-1根据帧率可以计算一个速度，得到一个(2,40)的速度向量，经过fc(2->16,relu)得到(16,40)的特征向量。

对于其他移动物体对目标物体的影响。不知道作者是怎么做的，感觉他们是联合预测，但是kitti上根本没有那么多有60帧的物体，这里我只能把需要预测的物体的60帧内的其他标记的物体位置都记录下来，然后一起预测了。虽然有些是行人，有些是静止的车辆。不过接下来的social pooling会过滤掉一些东西。

Social pooling过程比较复杂，对于i这个物体，取Y\_i,t的坐标为圆点，画一个半径为4m的圆，将圆内其他移动物体的坐标按照角度6个和距离6个，划分成6\*6的区域，于是可以构造出一个6\*6的矩阵，里面的值是gru上一层输出的隐状态h\_j,t-1（48维的向量），于是得到6\*6\*48的矩阵，这里如果遇到两个物体在同一个区域，就取平均。得到这个6\*6\*48的矩阵后经过全连接(6\*6\*48->48,ReLu)得到48维度的向量。

将上面三个向量拼接起来，得到一个16+32+48=96的特征向量，由于这是对每一帧的处理，所以一共是40个序列的特征向量，将这个序列放到GRU中（96,40）->（48,40），

对于最后一个序列输出的隐状态(48)，经过一个fc(48->(2,40),Relu)得到△Y(2,40),而每一层的H经过fc(48->1,linear)得到一个ranking分数。

这里需要注意的有三点。

1.对于每个物体，k个路线的产生是从CVAE来的，训练结果z=N(a,b),采样K次得到K个样本，后面就可以得到K个Yi，测试阶段直接采样Z=N(0,1),同样可以得到K个样本。

2.ranking的fc是共享fc，即所有序列都用同一个全连接参数，得到的ranking进行求和即是最后的得分。

3.最后输出的△Y是我们初始预测的（2,40）的Y轨迹的修正偏移量，用Ynew=Yold+△Y，所以后面部分可以迭代优化。

损失函数：

CVAE部分的损失：

评价损失的有两个，一个是对k个采样，经过优化后得到的Y‘，与真值比较得到平均的L2范数。

KLD损失：训练得到的zi和N(0,1)进行KLD，即求和(Z\_ki)log(Z\_ki)/(N\_ki)

Ranking和refine部分的loss：

交叉熵损失：

对于refine的Y’\_i,有Y’\_i\_old和Y’\_i\_new,他们的概率分布为softmax（-max||Y’i-Yi||）.这里max||Y‘i-Yi||其实是无穷范数，即取最大的偏差，经过softmax后得到两个概率分布p和q，计算loss=sum（-p(i)log(q(i))）

回归损失：

L=1/k\*sum||Yi-Y‘i\_new||

最后总的损失是这四个损失的和。

1. 《Intent-Aware Long-Term Prediction of Pedestrian Motion》

这篇比上一篇要早一年，用的是粒子滤波来预测。

这篇文章忽视了多智体间的互动，目标是预测一个长期的轨迹。这篇是基于目标预测的轨迹预测。首先假设目标是以很小的概率缓慢变化的。即p（gt+1|gt）表示一个目标变化概率。核心思想如下：

预测物体的下一个位置Xt+1受到当前位置Xt，目标Gt和环境参数wt影响，而下一个目标Gt+1是在Gt下进行变化的。这构成了一个MDP过程。地图上是用语义分为几类的，每类的奖励不同，优化目标是获取最大奖励。初始目标的产生是通过行人初始位置和朝向估计出来的。

这篇文章是一个很传统的轨迹预测，使用滤波来预测轨迹，比较的baseline是随机运动，常速运动，马尔科夫模型和离散模型。

1. 《Probabilistic Map-based Pedestrian Motion Prediction Taking Traffic Participants into Consideration》

这篇文章的核心是摘要的这句话：“与以前的工作相比，我们不仅考虑运动模型，语义图的约束和各种目标，而且还根据其他交通参与者的碰撞概率明确地调整了预测。”

1. 《R2P2: A ReparameteRized Pushforward Policy for Diverse, Precise Generative Path Forecasting》
2. 《INFER:INtermediate representations for FuturE pRediction》

这篇文章是到2019年的state of the art（在此之前的是DESIRE）。他的输入不是原始数据，而是语义信息。它把地图信息用语义分割成五个地图，分别是道路，行人道，障碍物，目标车辆和其他车辆。然后对这五个地图进行训练。