Discorso Poster

Introduzione

With the increasing use of autonomous driving and assistive technology, the importance of accurately recognizing and classifying objects around cameras and sensors has grown.
Unfortunately, the difficulties in this field are many.
In fact, it is hard for a NN to generalize over an unlabelled training set of 3D data, and doing manual labeling is unthinkable and infeasible.
The problem with autonomous technology is that even if it is trained to recognize a set, it must be trained to categorize unseen objects because they could appear in an open world.

Failed Attempts

To solve this problem, many researchers used **Zero-Shot Learning**, which can recognize unseen objects, using **side information** to transfer the knowledge from seen categories to unseen ones.
This method had great success in 2D
To use it in 3D it exploits:

- generative methods
- projection-based methods

The **generative method** trains a **fake feature generator**, supervised by seen classes and fine-tunes a classifier to discern real seen classes by synthetized unseen ones.
This method is useful in 2D, but has some limitations in 3D:

- 3D features more difficult to generate
- each time we see a new object, we have to restart the training

Used by 3DGenZ and SeCondPoint

The **projection-based method** aligns visual features to corresponding semantic ones, using seen class supervision.
In this way, the unseen classes can be recognized using similarities between visual and semantic features.
This method doesn’t retrain for each new unseen object but has limited performance.
Visual features can match, in fact, only a subset of the semantic features.
Used by TGP, which learns geometric primitives to facilitate the knowledge transfer from seen classes to unseen categories.

To solve this problem, cars use other sensors, such as lidars, to cover all the semantic features set.

Our model

We hence need a sensor fusion method to properly perform.
Our Multimodal zero-shot learning uses the image and point cloud **complementary information** for better visual-semantic alignment.
Considering that **images contain rich appearance features** and the **point cloud** possesses accurate location and **geometry features,** we propose **2 uni-modal complementary** visual data that generate more comprehensive visual features to align the semantic ones better.
Both seen and unseen classes will appear in the scene.
Our model will be a **GZSL**(can recognize seen and unseen objects) and **transductive** (there are present samples of unseen objects).

- ### Semantic-Guided Visual Feature Fusion (SGVF)

More flexible and applicable for zero-shot training
Semantic features have an **active role** in the visual feature fusion stage because they can choose the desired visual features to match, avoiding irrelevant interferences.
Better alignment.
In fact, visual features are generated to allow semantic features(The MLP network) to select effectively useful information to match semantic features for different categories of objects.

To avoid the **domain gap** between semantic and visual features, it is used Semantic-Visual Feature Enhancement **SVFE**.
The latter transfers the knowledge to the unseen class.

Methods

To train our model, we need

- all the possible frames of the point cloud,
- their corresponding image,
- the classes (split into seen & unseen)
- word embeddings (split into seen & unseen)

Execution

1. Feature Extraction:

   In this step, we pass from pure data to features and information.
   Extracts features from all the inputs.

   - To extract features from the **point cloud,** it's used **Cylinder3D**
   - To extract them from the **image**, we used **ResUnet**
   - To obtain **word embedding**, we used **W2V** & **Glove**
   - Then an MLP is used to extract semantic features from the word embedding

2. SVFE:

   In this step we have to reduce the gap between semantic and visual domains, to better align them.
   To do so, we make them interact with each other, enhancing the feature representation by **cross-attention mechanism**.
   If we only do SGVF alone, the gap between the visual and semantic features will obstruct the fusion.
   Cross-attention mechanism learn semantic visual projection automatically to enhance the most important features.

   - Semantic Enhancement

     It uses a **Transformer Decoder** $TD(q, k, v) = Linear(LN(MLP(Q) + Q))$
     with $Q = LN(CrossAttention(q, k, v) + q)$.
     Here q is F_s and works as a query, k is $F_l and\ F_i$
     The F_{es} enhanced semantic features is found as $F_{es} = TD(TD, F_s, F_l, F_l), F_i, F_i)$
     First point cloud features because we have to segment a point cloud.

   - Visual Enhancement

     Query to semantic features and fetching knowledge from semantic space.
     $F_{el} = TD(F_l, F_s, F_s)$ and $F_{ei} = TD(F_i, F_s, F_s)$

3. SGVF:

   Makes semantic features adaptively select valuable visual features from both modalities (image and point cloud) for effective feature fusion.
   How can we find good visual features for semantics?

   - Using weight matrices w_{3D} and w_{2D} generated by MultiHeadAttention respectively over (F_{es}, F_{el}) and (F_{es}, F_{ei}).

   MultiHeadAttention is designed to enable the model to focus on different parts of the input sequence simultaneously, thus capturing various aspects of the relationships between elements in the sequence. This mechanism improves the model's ability to understand context and dependencies in the data.

   - After this, element-wise multiplication between weight matrices and enhanced visual features is applied and the result is stored in a stack.

   - Finally, the fusion features are found by applying an MLP with softmax to the stack
     $F_{fusion} = MLP(softmax(stack(w_{3D} * F_{el}\ w_{2D} * F_{el})))$

   Now we finally have the fused features to transfer knowledge from seen to unseen classes.

4. Semantic-visual alignment:
   Transfers the knowledge from seen to unseen classes, by aligning fused visual features with semantic features, using side information (semantic features from word embedding).

- Loss function

  It uses **cross-entropy** loss and **unknown aware InfoNCE**.
  - L_S used to have compact distribution within classes and distinguishable one between classes.
  - L_u used to fight against the bias that pushes guesses to seen classes (because they are the only one that were used to train the model)
  - $L = L_S + L_u$

- Inference

  At the end for each point is chosen a class, determined by the **similarity** between fused visual features and semantic features of all classes(seen & unseen).

- Experiments

  We used as datasets **SemanticKITTI** and **NuScenes**

  Evaluation is done using hIoU and mIoU.

- Results

  0.097 seconds for each frame, so in can be used for real time computations
  - Comparison with 3DGenZ and TGP

  Some seen mIoU is better in other models, but the hIoU, which is the overall performance, is always better for our model.

  Furthermore, the mIoU of unseen classes have an improvement of 52% and 49% respectively to the datasets.

  TGP Identifies a motorcycle as a traffic sign(first column)and takes parts of a truck as a cyclist (third column).

  - Comparison with extensions of 2D methods

  3D Zero shot semantic segmentation has just got invented, so there are few comparison.

  To have other models to compare to, we adapted the 2D ones to 3D.

  The results are limited by the 3D more dimensions.

  - Comparison with multi-modal fusion methods

  The result is poor because features are fused directly without considering the semantic guidance.

- Ablation

  To verify if all the function used are useful.
  - without SGVF

    drop of 3,5%
  - without SVFE

    drop of 7%
  - only using LIDAR features

    drop of 7%