### Experiments & Results

#### II.b. The Partial Order Kernel

- **POKer accounts for the contributions of all the possible local alignments between any path in the query DAG and any path in the database DAG. For** $\beta \to \infty$, only the best alignments are taken into account.
- **POKer is computed using dynamic programming, with a linear time complexity with respect to the number of nodes in the strong product of input DAGs.**

#### Experiments on the Nordland dataset show that POKer effectively computes the similarity between DAGs, in an approach that is robust to appearance changes.

#### II.a. The Partial Order Kernel

- **POKer** is a convolution kernel for comparison of strings containing alternative substrings and represented by DAGs.
- Let $\Pi_x(G_x)$ and $\Pi_y(G_y)$ be the sets of paths of length $n$ in DAGs $G_x$ and $G_y$, respectively. POKer is defined as

$$K(G_x, G_y) = \sum_{n \geq 0} K_n(G_x, G_y) = \sum_{n \geq 0} \sum_{\pi_x \in \Pi_x(G_x)} \sum_{\pi_y \in \Pi_y(G_y)} \exp(\beta S(\pi_x, \pi_y))$$

- $S(\pi_x, \pi_y)$ is the score of the local alignment of $n$ nodes along path $\pi_x$ with the same number of nodes along path $\pi_y$. We use the same scores and gap penalty as in the POA.

- $\beta \geq 0$ is a parameter. Valid scores for it are those for which POKer is positive semi-definite.

#### I. Graph Representation of Image Sequences

- A set of **database** image sequences of the same place acquired at different times are aligned using the **Partial Order Alignment (POA)** algorithm (Lee et al., 2002) to obtain a DAG with branches. To find an optimal alignment:

  - a score is assigned to each aligned pair of nodes. We use the cosine similarity between image descriptors as scores. Nodes that are similar based on this score are merged.
  - gaps are penalized. We use a linear gap penalty.

#### Experiments & Results

- We evaluated our approach on the **Nordland dataset**, which consists of footage of a 728km-long train journey recorded once in every season. We subsampled each video at 0.5fps and cut its sequence into subsequences of length 15 to obtain 4 sets of image sequences: **Spring**, **Summer**, **Autumn** and **Winter**. We performed 4 experiments, each time using the sequences from a different season as the query DAGs and the triplets of sequences from other seasons as the database DAGs.

- Image descriptors were extracted from the **Places365-365** **Convolutional Neural Network** (Zhou et al., 2017), pre-trained for place recognition. Gap penalty and $\beta$ were set to -1 and 1, respectively.

- We used the methods presented in (Naseer et al., 2014) and (Milford et al., 2012), with the same descriptors as those used in our method, as baselines. We refer to them as **NetFlow** and **CNN+SeqSLAM**, respectively.

#### since the baselines compute similarity $\mathcal{S}$ of an image to another, not a sequence to a triplet of sequences, similarity $\mathcal{S}$ between query sequence $q$ and triplet of database sequences $\{s_1, s_2, s_3\}$ is computed as shown above.

#### II. The Partial Order Kernel

- **POKer** accounts for the contributions of all the possible local alignments between any path in the query DAG and any path in the database DAG. For $\beta \to \infty$, only the best alignments are taken into account.
- **POKer is computed using dynamic programming, with a linear time complexity with respect to the number of nodes in the strong product of input DAGs.**

### References

1. Abdollahyan and F. Smeraldi, POKer: a partial order kernel for comparing strings with alternative substrings in 2016 European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), 2017, pp. 283-286.