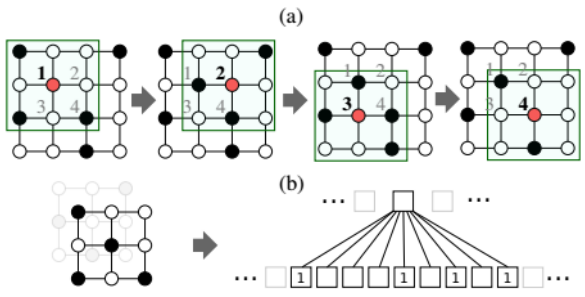
CNNs auf Graphen

30. November 2016

1 Einleitung

- Anwendungsfälle:
  1. Aus einer Menge von Graphen soll eine Funktion für Klassifizierungs- oder Regressionsprobleme gelernt werden, die auf nicht bekannte Graphen angewendet werden kann
  2. lerne Graph-Repräsentationen, um auf Graph-Eigenschaften (fehlende Kanten, Knoteneigenschaften) unbekannter Graphen zu schließen
- Graphrepräsentation:
  - Graphen können gerichtet oder ungerichtet sein
  - Graphen können zyklisch sein
  - Graphen können mehrere unterschiedliche Kantentypen besitzen (mehrere Perceptive-Field-Layer)
  - Graphen können mehrere diskrete oder kontinuierliche Werte an ihren Knoten haben
- Methode berechnet lokal verbundene Nachbarschaften der Graphen und benutzt sie als die *Receptive Fields* des CNN
- die Methode kann für Graphen mit gewichteten Kanten erweitert werden



- Idee: repräsentiere Bilder als Graph

  - ein Bild kann als Graph repräsentiert werden, indem die Knoten jeweils einen Pixel repräsentieren und es eine Kante zwischen zwei Knoten gibt, wenn deren Pixel benachbart sind
  - die lokale Nachbarschaft eines Pixels wird repräsentiert als ein Quadrat um den Punkt (hier 3×3)
  - Aus der Nachbarschaft kann ein Merkmal ermittelt werden
- üblicherweise gibt es keine räumliche Anordnung einer Graph-Repräsentation
- Probleme:
  1. Welche Nachbarschaften um welche Knoten und in welcher Reihenfolge bilden die Receptive Fields?
  2. Wie können die einzelnen Nachbarschafts-Graphen in einem Vektor repräsentiert werden (Normalisierung)?
- Verfahren:
  1. bestimme eine Knoten-Auswahl inklusive Reihenfolge
  2. bestimme den Nachbarschafts-Graphen um diesen Knoten mit genau k Knoten
  3. normalisiere die Nachbarschafts-Graphen
  4. füttere sie in ein CNN

2 Grundlagen

- Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und $E \subseteq V \times V$, wobei n Anzahl der Knoten und m Anzahl der Kanten
- Adjazenzmatrix A mit Größe $n \times n$, wobei $A_{i,j} = 1$, falls eine Kante von v_i nach v_j existiert (sonst 0) $\Rightarrow v_i$ und v_j sind adjazent
- ein Weg ist eine Sequenz von Knoten, bei der benachbarte Knoten adjazent sind
- $d(u, v)$ beschreibt die minimale Distanz zwischen von u nach v
- $N_1(v)$ beschreibt die 1-Nachbarschaft um einen Knoten, d.h. alle Knoten die adjazent sind zu v

2.1 Beschriftung und Partitionierung

- eine Graph-Beschriftung $l : V \to S$ bildet einen Knoten auf eine sortierbare Einheit ab
- induziert ein *Ranking* $r : V \to \{1, \ldots, |V|\}$ mit $r(u) < r(v)$ genau dann, wenn $l(u) > l(v)$
- falls l injektiv, dann gibt es eine totale Ordnung der Knoten in G und eine eindeutige Adjazenzmatrix A^l, bei der die Knoten die Position $r(v)$ haben
- eine Graph-Beschriftung induziert eine Partionierung $\{V_1, \ldots V_k\}$ mit $u, v \in V_i$ falls $l(u) = l(v)$

3 Lernen von Graphen

3.1 Knotenauswahl

- Auswahl an Knoten, für die ein Receptive Field erstellt werden soll
- Sortierung soll dem Verfahren von Bildern nahekommen, d.h. Knoten mit ähnlichen strukturellen Merkmalen sollen auch in der Vektorrepräsentation nah beieinanderliegen
- Graph-Beschreibung l – Metriken:
  - **Betweenness centrality**:
    - $g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$
    - σ_{st} beschreibt die Anzahl an kürzesten Pfaden von s nach t ist und σ_{st} die Anzahl dieser Pfade, die durch v gehen
  - **Eigenvector centrality**:
    - *Google's PageRank* ist eine Variante der Eigenvector centrality
    - $G = (V, E)$ mit Adjazenzmatrix A, sodass $a_{v,t} = 1$, falls eine Kante von v nach t existiert
    - relative Centrality von v: $x_v = \frac{1}{\lambda} \sum_{t \in N(v)} x_t = \frac{1}{\lambda} \sum t \in G a_{v,t} x_t$
    - kann als Eigenwertproblem formuliert werden: $Ax = \lambda x$
    - zusätzliche Einschränkung: alle Werte des Eigenvektors x sollen nicht-negativ sein $\Rightarrow$ bestimme größten Eigenwert $\lambda \Rightarrow$ eindeutig
  - **Degree centrality:**
    - Grad der Knoten, d.h. Anzahl adjazenter Knoten (gewichtet: Auswärtsgrad – Einwärtsgrad)
  - **Closeness centrality:**
    - durchschnittliche Länge zwischen dem Knoten und allen anderen Knoten
    - je zentraler ein Knoten ist, umso näher sind alle anderen Knoten
    - $C(x) = \frac{1}{\sum_y d(y,x)}$
    - kann sich für gerichtete Graphen stark unterscheiden (hohe Closeness für ausgehende Kanten, geringe Closeness für eingehene Kanten)
  - *Weisfeiler-Lehman Algorithmus*
  - *Page-Rank*
- es werden anscheinend mehrere Metriken benutzt, wie werden diese kombiniert?
- eventuell werden diese Metriken garnicht benötigt, da wir ja eine räumliche Struktur unseres Graphen besitzen!
- Gegeben: Graph-Beschreibung l, Abstand s, Anzahl w an Reciptive Fields

1. sortiere die Knoten auf Basis von l
2. iteriere über die sortierte Knotenmenge mit Abständen s, bis w Knoten ausgewählt wurden

3.2 Nachbarschaftssuche

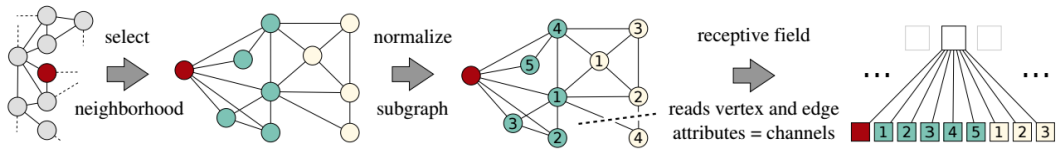
- Gegeben: Knoten v, Größe k des Receptive Fields

1. setze initiale Knotenmenge N auf v
2. wiederhole bis $|N| > k$:
   a) berechne für alle Knoten i in N die Nachbarschaften $N_1(i)$ und füge sie zu N hinzu

- Bemerkung: im Allgemein gilt $|N| \neq k$

3.3 Normalisierung

- Aus einem Nachbarschaftsgraphen soll ein Receptive Field konstruiert werden
- Knoten werden anhand eines Graph-Labelings l sortiert
  - ein Receptive Field für die Knoten (Größe k) und ein Receptive Field für die Kanten (Größe $k \times k$)
  - jedes Knoten- oder Kantenattribut wird in einem Receptive Field abgespeichert (z.B. Farbe)
- Gegeben: Menge von Graphen $\mathcal{G}$ mit k Knoten, Distanzmetriken für $k \times k$ Matrizen d_A und Graphen d_G für k Knoten
  - d_A, z.B. *Hamming-Abstand*: $d_A(x, y) = |\{j \in \{1, \ldots, N\} | x_j \neq y_j\}|$
  - Beispiel: 12345 und 13344 $\rightarrow 2$
  - d_G: z.B. *Edit distance*
- Optimierungsproblem über l: $\min_l \sum_{G \in \mathcal{G}} \sum_{G' \in \mathcal{G}} (d_A(A^l(G), A^l(G') - d_g(G, G')))$
- $\Rightarrow$ für beliebige Graphen G und G' soll die Ähnlichkeit dieser Graphen gleich der Ähnlichkeit der Graphen im Vektorraum sein (basierend auf den Adjazenzmatrizen der Graphen)
- $\Rightarrow$ Problem is NP-schwer
- Alternative: wähle aus einer Menge von Labelings die beste zu einer gegebenen Menge von Graphen
  - $\{(G_1, G'_1), \ldots, (G_N, G'_N)$ eine zufällge Auswahl an Graphpaaren von $\mathcal{G}$
  - wähle das Labeling l so, dass $\sum_{i=1}^{N} \frac{d_A(A^l(G_i), A^l(G'_i))}{N}$ minimal
- Labelings werden nur berechnet für Knoten gleicher Distanz zum Startknoten v
- Labelings sind im Allgemeinen nicht injektiv $\Rightarrow$ sortiere anhand lexikographischer maximaler Adjazenzmatrizen



4 Auswertung

- CNNs mit Bildern können identisch über CNNs mit Graphen dargestellt werden
- Methode funktioniert teilweise deutlich besser als State-of-the-Art Graph-Kerne (z.B. bei Klasifizierungsproblemen)

5 Zukünftige Arbeiten

- gewichtete Kanten (oder allgemeiner Graphen mit Kanteneigenschaften)
- Graphen auf andere Netze übertragen, z.B. RNNs
- kombiniere unterschiedliche Receptive Field-Größen