Proyectos Aprendizaje Computacional II - 2019

Eduardo Morales

May 15, 2019

Esta es una lista de proyectos para el curso de Aprendizaje Computacional II. Los proyectos pueden hacerse en equipos de máximo dos personas en cualquier lenguaje y sistema operativo. Si tienen algún proyecto en mente, también lo podemos considerar. Se van a realizar dos presentaciones de los proyectos. Una de avances, en donde se espera ver cómo van con sus proyectos y poderles dar retro-alimentación, y una presentación final en donde ya se va a calificar el proyecto. También se espera que entreguen un reporte al final del curso sobre su proyecto en formato de artículo tipo LNCS en inglés.

1) Estimación de pose: Recientemente han surgido algoritmos capaces de estimar la pose (esqueleto) de una o más personas a partir de videos. La idea del proyecto es re-implementar alguno de los algoritmos recientes para ésto y probarlo en videos en donde se ve sólo una parte del cuerpo, como el torso o sólo el brazo.

1. Se tiene que leer literatura reciente y ver cuál de los algoritmos existentes está disponible y/o es relativamente fácil de implementar. E.g., ver Realtime Multi-Person 2D Human Pose Estimation using Part Affinity Fields, CVPR 2017
2. Re-implementar el algoritmo y probarlo en bases de datos conocidas
3. Crear un conjunto de videos con información parcial del cuerpo y analizar la robustez del algoritmo

La idea es que al analizar posibles deficiencias de algoritmos actuales se puedan proponer, a futuro, algunas adecuaciones y extensiones para también detectar los movimientos de brazos robóticos.

2) RL-DL: Los algoritmos de aprendizaje por refuerzo se han combinado con algoritmos de Deep Learning para aprender a jugar juegos. El ejemplo más reciente es AlphaGo0 el cual se utilizó para jugar Go y posteriormente ajederéz. La idea es re-implementar está dupla (junto con MCTS) para aprender a jugar juegos sencillos. TensorFlow/Keras (DL) con OpenAI Gym (RL).

1. Entender cómo funciona AlphaGo0
2. Bajar e instalar herramientas de DL, como TensorFlow, y de RL, como OpenAI Gym
3. Realizar un algoritmo usando esta combinación en juegos sencillos y analizar su funcionamiento y desempeño

La idea es poder entender a detalle el funcionamiento de esta combinación y detectar sus limitaciones y posibles extensiones a futuro.

3) Ensambles de árboles de decisión: Los ensambles de clasificadores generalmente funcionan con algoritmos inestables, como los árboles de decisión, los cuales cambian sus modelos con pequeños cambios en los datos. La idea es crear un ensamble en donde se promueva la diversidad, forzando a los árboles a utilizar otros atributos.

1. Entender cómo funciona Random Forest
2. Dentro de la selección de atributos, en lugar de hacerla aleatoria dentro de un subconjunto, incluir una distribución de probabilidad acorde al uso que se tiene hasta el momento de los atributos en los árboles generados hasta ese momento
3. Pensar si es importante incluir información de atributos redundantes e irrelevantes

4. Hacer pruebas en varias bases de datos comparando Random Forest y el Random Forest diverso

4) Aprendizaje semi supervisado: Los algoritmos semi-supervisados en ocasiones no reportan los resultados esperados. Esto puede ser porque los ejemplos etiquetados y no etiquetados no provienen de la misma distribución o porque los modelos usados no son adecuados para representar los modelos. La idea es crear un algoritmo de aprendizaje semi-supervisado que detecte cuándo puede fallar y corregir o desechar a los ejemplos no etiquetados.

1. Implementar un algoritmo de aprendizaje semi-supervisado (lo más directo es usar *self-training*)
2. Crear un mecanismo que detecte de forma temprana cuándo el algoritmo no está funcionando
3. Cambiar el clasificador base y volver a probar

5) Datos desbalanceados: Los clasificadores muchas veces se ven afectado cuando existe un desbalance en las clases. Lo más común es realizar un sobre o sub-muestreo para balancear las clases, sin embargo el sobre o sub-muestreo sólo se debe de realizar mientras no se tengan resultados aceptables. La idea es hacer un algoritmo de clasificación con datos desbalanceados, en donde se haga sobremuestreo sólo si es necesario, y hacerlo sólo para que se tengan resultados aceptables (i.e., no se tienen porqué balancear las clases)

1. Implementar un algorimo de balanceo de clases (puede ser ROS, RUS, Smote, etc.)
2. Idear un mecanismo que detecte la cantidad de muestro a realizar con base en unos cuantos experimentos
3. Conjuntar este mecanismo en un algoritmo y probarlo contra otros algoritmos que realizan o no muestreo.

6) Aprendizaje multi etiqueta: Los algoritmos de clasificación multi-etiqueta tratan de predecir el valor de varias clases al mismo tiempo. Una forma de hacerlo es mediante clasificadores en cadena, incorporando clases a los modelos gradualmente. Una mejora a esto, es estimando el orden en el cual hacer este encadenamiento. La idea del proyecto es no sólo determinar el orden, sino también los atributos relevantes dentro de cada clasificador.

1. Bajar Meka (versión de Weka para clasificadores multi-clase)
2. Usar de base BCC (propuesto en el INAOE :-)) para detectar el orden de las clases
3. Usar un algoritmo de selección de atributos de Weka e introducirlo dentro del algoritmo de BCC
4. Realizar pruebas con BCC y otros clasificadores multiclase

7) ILP: Los algoritmos de ILP son capaces de encontrar modelos que expresan relaciones entre objetos y que pueden incluir entre otras cosas, variables y funciones. Esta expresividad adicional les permite atacar problemas que son difíciles para otras técnicas de aprendizaje. Recientemente se han extendido estos modelos relacionales para incluir información de probabilidad, juntando dos de los principales enfoques de representación: lógica y probabilidad. El proyecto consiste en re-implementar un sistema de aprendizaje relacional estadístico reportado recientemente.

1. Bajar, instalar y probar DC, extensión de ProbLob para Prolog con probabilidades de distribuciones continuas
2. Leer el algoritmo Learning Relational Affordance Modelos for Two-Arm Robots, IROS 2014.
3. Re-implementar lo reportado en el artículo

Identificar problemas y dar sugerencias de mejoras para este trabajo.

8) Perfilado de usuarios en Twitter: A partir de los tweets de usuarios mexicanos, desarrollar métodos que permitan determinar la profesión del usuario y posiblemente otros aspectos (lugar de residencia, etc.) a partir de la información textual y visual en tweets.

1. Desarrollo de métodos para el análisis de la información textual.
2. Estudio e implementación de descriptores visuales para representar el contenido de las imágenes (e.g., prescencia de caras, interior/exterior, esquinas, texto, etc.)
3. Implementar métodos de clasificación (posiblemente multi-etiqueta) que combinen información visual y textual (posiblemente ensambles).
4. Evaluación.

9) Macros: Los macros o macro-operadores permiten reducir la profundidad en la búsqueda de la solución de problemas, pero aumentan su arborecencia. El proyecto consiste en diseñar un sistema de aprendizaje de macros que incorpore un mecanismo de filtrado para conservar sólo los macros que presenten beneficios al sistema.

1. Implementar un sistema que permita aprender automáticamente macros generales que puedan aplicarse en varios estados
2. Diseñar un sistema de búsqueda que incorpore estos macros dinámicamente
3. Definir un esquema de filtrado para eliminar macros poco útiles
4. Probar el esquema en más de un dominio
5. Compararse con trabajo relacionado

10) Causal: Recientemente se han desarrollado modelos para capturar información causal. Se espera que el conocimiento causal sea de gran relevancia en el área de robótica. Este conocimiento causal puede utilizarse de diferentes formas, por ejemplo, como restricciones para o complemento de un sistema de aprendizaje por refuerzo.

1. Implementar un sistema que permita aprender un modelo causal a partir de datos
2. Incorporar ese modelo causal como conocimiento adicional en un sistema de aprendizaje por refuerzo
3. Realizar pruebas del sistema con/sin el modelo causal