Trabajo Final

Seminario
Gramáticas formales: formalismos e implementaciones

Departamento de Letras
FFyL - UBA
Junio de 2022

Tal como figura en el programa, para aprobar el seminario, quienes hayan regularizado deben presentar un trabajo final. Este trabajo final debe abocarse a una (y solo una) de las siguientes opciones:

- **Opción 1**: Una monografía de entre 8 y 12 paginas (entre 4000 y 6000 palabras).
- **Opción 2**: Reseña crítica de entre 6 y 10 páginas (entre 3000 y 5000 palabras) de algún parser o recurso computacional que involucre gramáticas y formalismos gramaticales.
- **Opción 3**: Una contribución computacional de acceso público, que puede ser una mejora a partir del *fork* de alguna librería previamente existente o puede ser un desarrollo propio.

La fecha límite para la entrega de trabajos es la que prevea la reglamentación de la facultad. De todos modos, en la medida de lo posible, se agradecerá que no demoren la entrega. No serán recibidos trabajos una vez que venza el plazo que la facultad disponga. Sientanse libres de contactarnos ante cualquier duda a nuestros mails:

fernandocarranza86@gmail.com, pablo.zd@gmail.com,
m.fernandezurquiza@gmail.com, juliamilanese@gmail.com

Debajo especificamos más detalles de la modalidad de cada una de las opciones.

1. Opción 1: Una monografía

Puede optarse por escribir una monografía en la que se defienda una postura original acerca de algún problema que involucre alguno de los temas vistos durante la cursada. La monografía debe tener entre ocho y doce páginas (entre 4000 y 6000 palabras). A modo ilustrativo, estos pueden ser algunos de los posibles temas para un trabajo monográfico acorde a los temas vistos en el seminario, si bien esta lista es, desde ya, no exhaustiva:

1. Comparación del modo en que dos o más modelos teóricos vistos analizan un fenómeno gramatical y síntesis superadora o defensa de algún análisis en particular.

2. Discusión que involucre alguno de los marcos teóricos vistos.

3. Problematización acerca de cuál es o debería ser la relación entre las implementaciones computacionales y la lingüística teórica.

Recomendamos que, en cualquier caso, nos consulten por el tema elegido antes de comenzar la escritura.

2. Opción 2: Reseña crítica

La reseña crítica tiene que abordar alguna librería de NLP que procese sintácticamente oraciones. Debajo incluimos un listado no excluyente de algunas de las librerías o recursos computacionales que podrían considerar:

1. Freeling (https://nlp.lsi.upc.edu/freeling/node/1)

2. OpenCCG (https://www.nltk.org/api/nltk.ccg.html)

3. ACE (no visto durante la cursada: https://pydelphin.readthedocs.io/en/latest/guides/ace.html)

4. MaltParser (https://maltparser.org/)

5. SpaCy (https://spacy.io/)

6. NLTK (https://www.nltk.org/)

7. Parsers desarrollados por Stabler (https://github.com/epstabler/mgtdb)

8. XTAG (no visto durante la cursada: https://www.cis.upenn.edu/~xtag/)

9. Stanza (https://stanfordnlp.github.io/stanza/)

La reseña tiene que tener una estructura clara. Si bien existe siempre la posibilidad de introducir variantes, se sugiere seguir a grandes rasgos la siguiente:

1. **Introducción**: Inicialmente, identificar el recurso o librería que se va a discutir (indicar, en caso de corresponder, institución, autores, link al repositorio, fecha de publicación y otros detalles que se consideren importantes) e indicar los objetivos de la reseña, detallando los puntos principales que se van a discutir.

2. **Resumen**: Resumir los objetivos y principales características del recurso computacional, cómo funciona y qué clase de resultado de devuelve.

3. **Discusión**: Hacer un análisis crítico de la librería o recurso, contextualizándola, caracterizándola y evaluando su rendimiento y sus principales ventajas y desventajas.

4. **Conclusiones**: Hacer un cierre con las principales ideas que se sostuvieron a lo largo de la reseña.

La reseña puede estar dividida en secciones y subsecciones o estar concebida como un texto único sin divisiones internas. Dado que se trata de un texto corto, sugerimos la segunda opción.

Aquí reproducimos una selección de recomendaciones para chequear tomadas y adaptadas de las normas para la escritura de reseñas de la Revista RASAL:

1. Todo el texto de la reseña debe estar escrito con un tono evaluativo y expresar una toma de posición ante la librería reseñada, aunque no necesariamente laudatoria de cada uno de los aspectos de la misma.

2. Se debe hacer explícita al comienzo del texto de la reseña la tarea central que resuelve la librería.

3. Contextualizar la librería en el marco de otras librerías semejantes y establecer una breve comparación.

4. Se debe explicar cómo funciona la librería, cuáles son sus métodos principales y cuáles son sus ventajas.

5. Identificar las limitaciones del alcance de la librería y sus puntos débiles. Esto es un rasgo constitutivo del género reseña.

6. Utilizar fuentes bibliográficas relacionadas con la presentación de la librería.

7. Al final del cuerpo de la reseña, consignar el listado de referencias bibliográficas encabezado por el título "Referencias".

8. Se deben seguir, en todo momento, convenciones de cita y formato convencionales.

3. Opción 3: Una contribución

La segunda opción para el trabajo es realizar un aporte mediante la escritura de algún recurso computacional en Python. Este aporte puede ser una contribución enteramente original o una contribución a una librería ya existente. En este último caso, para la entrega y aprobación no es necesario que la contribución sea enviada o aprobada por quienes mantienen la librería original. Este trabajo puede ser individual o grupal, con un máximo de tres miembros por grupo. La elección de hacerlo en grupo debe estar justificada por el volumen de la contribución y todos los integrantes del grupo deben contribuir con código al repositorio.

A continuación hacemos un listado no excluyente de posibles ideas para contribuciones:

1. Extender el sistema de Warstadt (`https://github.com/alexwarstadt/minimalism`) de modo tal que pueda aplicarse para parsear oraciones.

2. Revisar el parser de dependencias de NLTK y hacer que sea capaz de limitar la cantidad de argumentos.

3. Contribuir con un parser de dependencias más cercano a la propuesta de Gaifman presentada en clase.

4. Hacer un algoritmo que traduzca las gramáticas basadas en constituyentes que usa NLTK en gramáticas basadas categoriales y/o viceversa.

5. Agregar al parser para gramáticas categoriales de NLTK la posibilidad de definir categorías derivadas que incluyan rasgos.

Habitualmente, los repositorios de las librerías tienen una sección denominada *Issues*, donde los usuarios pueden ir dejando comentarios sobre mejoras o arreglos que es necesario realizar en la librería en cuestión. Estos comentarios también pueden resultar una buena fuente de inspiración, siempre y cuando sean pertinentes con las temáticas abordadas durante el seminario.