

Título del Trabajo  
Fin de Grado

Grado en Ingeniería Informática



Trabajo Fin de Grado

Autor:

Nombre y Apellidos

Tutor:

Nombre y Apellidos

Mayo 2024



# Resumen

El resumen, según normativa, debe tener un máximo de 500 palabras. Tiene que estar en la lengua en la que entregues la memoria (cualquiera de los oficiales: valenciano, castellano o inglés) y puede tener además una versión en inglés. Tiene que ser el mismo que envíes cuando pidas presentar. Se hace al final del trabajo y es un resumen de todo.

Observa que entre la portada y la primera página siempre hay que dejar una página en blanco. De esa manera al abrir el libro aparece directamente el texto en la parte derecha. Tenlo en cuenta en esta sección o en las siguientes si quieres obtener ese efecto (puedes verlo en el “Print Preview” del editor que uses).

Se añade a veces, justo después de la portada una especie de portada interior dónde se repite el título, autor, tutor, logos de la universidad, etc. Se hace también por estética, para que nada más abrir la portada aparezca esta.

También en ocasiones hay una página antes del resumen que es un “Preámbulo” que sirve como una pequeña explicación de la razón de ser del trabajo. Algo así como una respuesta a la pregunta “¿qué me motivó a abordar este tema?”. No es muy habitual, pero si por alguna razón sientes que la tienes que añadir, hazlo sin problemas. Ocuparía máximo una página.

# Agradecimientos

Sección opcional (aunque es bastante habitual) en el que envías agradecimientos a quién quieras. Suele tener máximo 1 página.

# Citas

Opcional también. Se puede incluir una cita o más. Máximo una página y justificado a la derecha.

Índice de contenidos

[Resumen 2](#__RefHeading___Toc6149_2675187765)

[Agradecimientos 3](#__RefHeading___Toc6151_2675187765)

[Citas 4](#__RefHeading___Toc6153_2675187765)

[Índice de figuras 6](#__RefHeading___Toc6155_2675187765)

[Índice de tablas 7](#__RefHeading___Toc6157_2675187765)

[1. Introducción 8](#__RefHeading___Toc6159_2675187765)

[2. Estado del arte. 10](#__RefHeading___Toc6171_2675187765)

[3. Objetivos 11](#__RefHeading___Toc6175_2675187765)

[4. Metodología 12](#__RefHeading___Toc6177_2675187765)

[5. Análisis, especificación y diseño 13](#__RefHeading___Toc6179_2675187765)

[6. Implementación 14](#__RefHeading___Toc6199_2675187765)

[7. Pruebas y validación 15](#__RefHeading___Toc6201_2675187765)

[8. Resultados 16](#__RefHeading___Toc6203_2675187765)

[9. Conclusiones y trabajo futuro 17](#__RefHeading___Toc6205_2675187765)

[Referencias 18](#__RefHeading___Toc6207_2675187765)

[Apéndice I 19](#__RefHeading___Toc6209_2675187765)

[Acrónimos 20](#__RefHeading___Toc6209_26751877651)

# Índice de figuras

# Índice de tablas

# Introducción

Antes de empezar, es importante explicar que esta plantilla fue creada para ayudar a empezar a escribir el TFG, pero ni es una imposición, ni debe tomarse como una algo oficial. En general, antes de enviar la memoria de tu TFG es importante que repases que se cumple con toda la normativa (especialmente el libro de estilo) y no fiarte de esta plantilla (porque es vieja y puede que haya cambiado algo). La normativa la encontrarás en la web de la asignatura.

Aquí habrá una serie de explicaciones sobre cómo funciona una memoria de un TFG genérico que he subrayado en amarillo. Todo lo amarillo por tanto tendrá un carácter general y podría no adaptarse perfectamente a tu TFG. Sin embargo hay otras notas que he personalizado especialmente para ti y que he subrayado en rojo (o más bien rosa) y que sí son específicas de tu TFG.

Tanto unas como otras, amarillo y rojo, por supuesto se deben eliminar y se deben sustituir por el texto correcto (o puedes crear un nuevo documento y trabajar directamente en él o lo que creas conveniente).

-------

Justo después de los índices añadir la introducción. Acuérdate de actualizar los índices antes de enviar la versión definitiva (es un error habitual). Tiene que haber índice general, de tablas y de figuras. Para que se generen correctamente y todo sea más fácil es importante usar referencias cruzadas. Usa también los estilos y demás elementos de ayuda que proporciona el procesador de textos. Usarlo bien desde el principio te va a ahorrar tiempo justo al final, cuando vas a ir más estresado.

En cuanto a la introducción, explica el contexto en el que se va a desarrollar tu trabajo teniendo en cuenta que el lector puede no tener ni idea de lo que estás hablando. Piensa en una persona que no tiene ni idea de Informática y explícaselo así.

Dentro de la introducción, puede haber secciones como “estudio de viabilidad” (DAFO), “marco teórico”, etc. Estos puntos, dependiendo de la importancia que les queramos dar pueden ir como subapartados de la introducción o al mismo nivel.

En ocasiones, también se pone un pequeño punto con las motivaciones (en lugar de hacerlo en el “Preámbulo” que te comenté anteriormente).

Lo que es muy importante es que haya un punto de “justificación” del TFG. Este puede ser un subapartado de la introducción o puede ser un apartado al mismo nivel, como prefieras.

Todas las afirmaciones que hagas en esta sección deben estar justificadas por la bibliografía (p.e. si dices “tal algoritmo es más rápido que tal otro” debes añadir una cita a la bibliografía que lo justifique). Utiliza bibliografía de la mejor calidad posible (mejor publicaciones científicas que blogs o webs random que encuentres).

Y tanto en esta sección como en cualquier otra, es mejor utilizar la tercera persona ya que otorga un aire de objetividad y formalidad al trabajo. Al utilizar expresiones como "se observa", "se concluye" o "el estudio demuestra", se enfatiza la validez de los resultados y se minimiza la influencia de la opinión personal del autor. Además, la tercera persona sugiere que los hallazgos son aplicables a un contexto más amplio y no están limitados a la experiencia personal del autor. Esto fortalece la generalización de los resultados y aumenta su relevancia. Quizás en las únicas secciones en las que se puede aceptar la primera persona serían precisamente en la introducción y en las conclusiones, pero aún así, es mejor usar la tercera.

En tu TFG en concreto, vas a implementar una aplicación de escritorio para realizar búsquedas utilizando **lenguaje natural**. Así que yo empezaría hablando de esos temas:

* Panorama actual (limitaciones de los metadatos tradicionales para la recuperación de información, desafíos técnicos [falta de precisión] y éticos [sesgos, alucinaciones…], etc)
* Lenguaje natural, visión artificial y multimodalidad (avances en inteligencia artificial y procesamiento del lenguaje natural, grandes modelos del lenguaje [LLM], qué es la visión artificial, qué significa multimodalidad, modelos cuantizados, modelos ligeros (lightweight models), etc). En cuanto al nivel de profundidad, elige hasta dónde quieres llegar: puedes quedarte muy en la superficie y solo tratarlo a nivel de aplicaciones y librerías o te puedes bajar a un nivel más teórico y profundizar hasta redes neuronales o incluso explicar la neurona artificial. Cualquier opción es buena, lo importante es que el tema siempre esté centrado en el propósito del TFG.
* Visión artificial y multimodalidad (qué es cada uno de estos elementos, cómo funciona CLIP, etc)
* Sistemas RAG (que son, cómo funcionan, etc)

Y tras esos puntos, ya tendrías todo listo para un punto de “justificación” [del proyecto]. La justificación en un trabajo de este tipo debería aparecer siempre.

# Estado del arte.

Esta es la primera sección que debería empezar a desarrollarse. Eso no significa necesariamente que deba estar escrita desde el principio y sea inmutable. Lo importante es que condensa el nivel de desarrollo de la investigación en Informática en este momento, cuales son las últimas innovaciones en el campo en el que estamos trabajando. Por eso, esa investigación sobre qué es último se hará al principio del TFG, las primeras semanas y va a ser la principal fuente de referencias bibliográficas de tu trabajo, porque cada vez que leas algo y aprendas de él o lo vayas a añadir de alguna manera al trabajo final, lo deberás citar. Intenta siempre que las referencias sean de sitios con buena reputación, especialmente revistas científicas (journals) y no añadir muchas referencias a blogs o sitios de Internet. Sería conveniente usar un software gestor de referencias para este trabajo. También tienes que decidir desde el principio (aunque si usas un gestor de referencias se puede cambiar después), qué modelo de citas vas a usar. A mi me gusta APA 7, pero cualquiera bien usado está bien.

Es muy importante estudiar el estado del arte de todas las partes de tu trabajo que lo integran y es un error común no hacerlo y descubrir tarde que había alguna posibilidad mejor y difícil de integrar a última hora. Por ejemplo, si tenemos un trabajo acerca de la detección de cierta enfermedad con imágenes de rayos X, habrá que investigar sobre la enfermedad, sobre rayos X y sobre algoritmos de visión artificial. Cada una de las investigaciones será un apartado e incluso podría haber un apartado en específico sobre la convergencia de 2 de estas investigaciones (p.e. visión artificial con rayos X). Si además hay literatura sobre la convergencia de todas, es posible que no estemos realizando un trabajo demasiado original y habría que buscar en qué sentido lo es y explicarlo bien.

Este apartado va a dividirse por secciones que dependen mucho de la temática en concreto del TFG. Algunas serán específicas del tema como se comentaba en el párrafo anterior, pero otras secciones comunes son “Antecedentes”, “Contexto”, “Desafíos encontrados”, “Problemática y limitaciones”, “Líneas de investigación emergentes”… Deberás decidir si añadir unas u otras.

Este apartado es fundamental en un trabajo como el tuyo que tiene cierta carga de investigación. En tu caso, se trataría de investigar cuales son los modelos de deep learning que hay hoy en día capaces de funcionar en sistemas con un hardware estándar para un usuario de ordenadores. Se puede dividir la sección en modelos de lenguaje, de visión, audio, multimodales… lo que pruebes y consideres que sea interesante contar al mundo.

# Objetivos

Es fundamental definir claramente los objetivos del TFG. Hazlo con verbos de objetivos y dejando clarisimo cuales son los objetivos principales y los secundarios. El punto de objetivos puede ser una sección de la introducción, aunque se le da más importancia si tiene un apartado al mismo nivel que esta.

Un error común es confundir los objetivos del TFG con las funcionalidades de una aplicación. Las funcionalidades se desarrollarán más adelante, pero los objetivos que se plantean en este punto son lo que queremos obtener al iniciar el proyecto (“desarrollar un programa”, “aprender tal tecnología”, etc).

Es de las primeras secciones que se deberían elaborar y esta no debería modificarse mucho entre el inicio del TFG y el final. Los objetivos nos indican cómo de ambicioso es el proyecto y por tanto definen su alcance.

Esta sección suele tener un par de páginas.

En la presentación de diapos tienes el objetivo principal, solo es copiar y pegar y luego explicar un poco más sobre lo dicho. Luego añade algunos objetivos secundarios que prácticamente se sacan del enunciado.

# Metodología

¿Cómo se va a desarrollar el proyecto? En principio, se debería explicar cómo se va a trabajar, tanto desde un punto de vista de organización (metodología de desarrollo, como RUP, SCRUM, XP, etc), como técnico (herramientas que se han elegido, como lenguajes de programación y librerías importantes, hardware, etc).

Es una sección viva que se puede empezar a desarrollar al principio, pero que va a ir evolucionando con el transcurso de los meses.

Suele ser un apartado bastante largo, que a veces se divide incluso en puntos al mismo nivel, según la importancia que se le quiera dar a cada parte. Suele dividirse en apartados como “Herramientas” y/o “Tecnologías” (que a su vez pueden dividirse en secciones para cada una de ellas), “Hardware” o incluso puntos específicos del tema elegido para el TFG (por ejemplo, si se está entrenando un modelo de deep learning, puede haber un apartado “Preparación del dataset”).

Puedes enfocar la metodología desde dos puntos de vista:

1- Cómo vas a trabajar (y contar la forma en la has estructurado el trabajo: RUP, metodología ágil, etc) [esto seria darle más importancia al método de trabajo]

2- Cómo has realizado la investigación para llegar al resultado final [esto sería darle más importancia al método de investigación]. En este caso no hay que volver a explicar lo que has explicado en el estado del arte, en la introducción o en otras secciones; lo que se espera de una metodología de un trabajo de investigación es que se explique cuanto de exhaustiva fue la búsqueda (por ejemplo, en tu caso, ¿cómo encontraste los LLM que has probado? ¿cuales fueron tus criterios para probar uno y no otro? Etc).

También puedes explicar las dos cosas. Lo que prefieras.

# Análisis, especificación y diseño

Dependiendo del proyecto se puede llamar de una manera u otra (y también según la importancia que le quieras dar a cada concepto). Por ejemplo podría llamarse simplemente “Análisis” o “Análisis y diseño”. Si el proyecto se hace con una metodología ágil como SCRUM, el análisis podría ser el resultado del sprint 0 en el que se ha creado un backlog con todas tareas a realizar.

En ocasiones esta sección se omite y se inserta en otra sección como subapartado (por ejemplo, en metodología o en Desarrollo o directamente se une como una sección “Análisis y desarrollo”). La elección debería ir en función del proyecto en concreto. Ciertos profesores aprecian que exista por separado una sección de análisis y otra de desarrollo.

En general, la idea de esta sección es obtener requisitos (funcionales y no funcionales) que pueden estar tanto en el formato IEEE 830 como en los formatos habituales de las metodologías ágiles (Historia de Usuario, por ejemplo). Es importante que cada uno de estos elementos tenga su identificador para que luego desde el resto de secciones. Lo trascendental aquí es que todo esté cohesionado, que no haya requisitos que luego no se implementan, ni funcionalidades que se implementan sin que un requisito lo solicite.

En cuanto al diseño, es importante recordar que no se trata solo de diseñar interfaces de usuario, sino que hay que diseñar toda la solución a nivel interno. Esto requiere muchas veces la creación de diferentes gráficos UML (aunque también pueden existir gráficos más visuales, fuera del estándar) que **deben explicarse**. También se deben diseñar las bases de datos si esto es importante en tu TFG.

Si el apartado de diseño tiene muchos elementos que ameritan crear sub-apartados, sería mejor darle entidad propia de primer nivel. Secciones habituales podrían ser “Diseño del API”, “Diseño de la arquitectura”, “Diseño de la base de datos”, “Diseño del front-end/back-end”, “Diseño de interfaces”, “Diseño de test unitarios”…

No hay mucho que explicar aparte de lo ya dicho: todo lo que hayas hecho para diseñar la APP ponlo aquí. No pierdas el tiempo haciendo diagramas, mockups, etc una vez desarrollado el programa, si no lo has usado para el análisis previo al desarrollo es superfluo añadirlo después. El análisis y diseño sirve para ayudarte en el desarrollo, no está para rellenar. Yo por ejemplo cuando tengo que diseñar un interface, no lo hago con una herramienta de mockups, lo hago en una hoja de papel y luego directamente va al diseño en el lenguaje que sea (y más ahora que existen herramientas de IA que de una foto de un papel te sacan un interface).

# Implementación

Esta sección también se puede llamar “Desarrollo”, pero también a veces se integra junto al “análisis y diseño” y otras veces junto a los resultados. Evidentemente, siempre hay una parte de implementación en todo TFG, pero a veces es menor o menos importante (por ejemplo, en TFGs que tienen mucha investigación) y por eso se *esconde* un poco.

Es importante que tengas en cuenta que lo que se espera de esta sección es que expliques el proceso de desarrollo, no que pongas un listado del código. Solo pondrás listados de código de elementos puntuales o algoritmos muy importantes. Si el listado es muy importante, pero muy largo es mejor que lo añadas como un anexo y aquí destaques solo algún elemento puntual. Si has seguido SCRUM modificado como metodología, aquí se debería mostrar una tabla o una figura que indicara tras cada sprint qué producto se tenía (qué funcionalidades implementaba). Esta, por tanto, es una sección que se irá rellenando conforme avance el desarrollo (y te recomiendo que así sea y no lo dejes para el final porque podría ser que al final no te acuerdes de ciertos detalles).

También se pueden explicar partes complejas o problemas que hayas encontrado. No hace falta que expliques cosas sencillas o rutinarias. Por ejemplo, si para gestionar tu proyecto has usado GitHub, no hace falta que expliques el proceso de alta o cómo hacer un *commit*. Sin embargo, sí que sería interesante que explicaras cómo has configurado un Docker si tu proyecto los usa. La línea entre lo trivial y lo importante la marcas tú, pero como idea utiliza la siguiente pregunta: ¿un ingeniero informático recién licenciado sabría hacer la tarea si solo se la indico en un comentario informal en un pasillo?

Si no hay una sección de Resultados en tu memoria del TFG, se debería añadir aquí un subapartado para mostrar cómo ha quedado tu proyecto (interfaces o los ficheros que genere). En cualquier caso, esta sección se debería dividir en las partes de entidad suficiente que hayas desarrollado (podrían ser “API de tal cosa”, “Servidor de esto”, “Dataset”, “Base de datos”, “Despliegue”, “Pruebas”, etc), dependiente totalmente del tema de tu TFG.

Por el tipo de TFG que tienes entre manos, esta sección no es la más importante y podría estar embebida entre los resultados. Pero como lo veas.

# Pruebas y validación

Habitualmente esta sección se mete dentro de “Implementación”, pero si es muy importante, se puede añadir como un punto separado. Básicamente se trata de mostrar cómo se comporta la aplicación o proyecto creado y demostrar de alguna manera que no tiene fallos. También se deberían mostrar los objetivos que te has marcado en cuanto a requisitos no funcionales (por ejemplo, en el caso de un proyecto web, si hay un RNF para que soporte 1000 usuarios simultáneos, deberías demostrarlo con un test de carga mostrando los tiempos de acceso para el usuario 1000).

Por supuesto, hay veces que hay fallos que no da tiempo a pulir. Si los hay no hay que esconderlos, sino afrontarlos, comentarlos, explicar porque creemos que podrían estar ocurriendo y proponer soluciones que se implementarían en un futuro.

Esta sección en tu caso la fusionaría con los resultados porque lo más interesante sería mostrar resultados cualitativos y no tanto cuantitativo. Me refiero a mostrar qué se puede hacer con algunos ejemplos reales de búsquedas y de etiquetado de ficheros. No hace falta mostrar datos como el número de ficheros por unidad de tiempo que se pueden analizar o cosas así (que si da tiempo al final y quieres, puedes hacerlo, claro).

# Resultados

Esta sección debería aparecer siempre, especialmente en TFGs con mucha investigación. De lo que se trata aquí es de mostrar resultados finales. Los interfaces ya se mostraron en la sección de Implementación, por lo tanto aquí no hay que mostrarlos. Aquí se trata más de mostrar otro tipo de resultados: los resultados de negocio, o dicho de otra forma, en qué va a mejorar al usuario la existencia de este proyecto (especialmente en comparación con todo lo que había hasta el momento).

Hay ciertos proyectos que se pueden probar con usuarios reales (testers), que pueden rellenar una pequeña encuesta de satisfacción y hacer algunos comentarios que se pueden analizar cualitativamente. Ese podría ser un elemento muy interesante a incluir en esta sección si diera tiempo a desarrollarlo. Pero en cualquier caso, hay una serie de datos que todo proyecto ofrece como resultado: se trataría de hacer una especie de demo para un determinado caso de uso en concreto.

Esta sección que es de las últimas que se desarrollará, debería tener cierta correspondencia con la sección “Objetivos”, ya que los resultados deberían mostrar que se cumplieron todos los objetivos marcados.

Como te he comentado en la sección anterior, aquí lo interesante es mostrar que la aplicación hace lo que se espera con ejemplos reales.

# Conclusiones y trabajo futuro

Las conclusiones son muy importantes y deberían tener una sección de primer nivel siempre. Habitualmente esta sección une las conclusiones con los trabajos futuros, pero también se podrían separar (aunque “Trabajos futuros” habitualmente es un sub-apartado de otro elemento). También en ocasiones hay una sección de primer nivel o unida a “Conclusiones” que se denomina “Discusión” en la que se comparan los resultados de nuestro trabajo con el de otros (aunque esta sección es más habitual en trabajos fuertemente orientados a la investigación).

La sección de Conclusiones no suele ser excesivamente larga, 3 o 4 páginas, aunque va a depender de si metemos otras secciones dentro o no. Se puede dividir en sub-apartados o no. Si se divide, podría tener sub-apartados como “Problemas afrontados”, “Lecciones aprendidas”, “Limitaciones” y, por supuesto, “Trabajos futuros”. En “Trabajos futuros”, se debería explicar cúal sería la línea de trabajo para corregir o solucionar los problemas y limitaciones que tiene la solución que se ha presentado. Así mismo, también se podría imaginar qué nos gustaría en un futuro añadirle al proyecto más allá del alcance que habíamos fijado al inicio del TFG.

Si el proyecto iniciado se quiere comercializar de algún modo (o abrirlo tipo *open source*), también se debería explicar en esta sección y presentar un plan para hacerlo realidad. Esto suele valorarse muy positivamente por los tribunales.

Aquí intenta no personalizar demasiado, no hables en primera persona (error habitual). Añade todas las features que se te ocurran que podrían mejorar el proyecto.

# Referencias

Todas los artículos, libros, páginas web, vídeos, etc. que hayas usado para aprender algo o justificar algo en tu TFG deben tener una referencia junto con su correspondiente cita. Es una sección que se va construyendo casi automáticamente con el desarrollo y que es muy importante para ver que el alumno no se ha inventado ni plagiado sin citar aquello que ha dicho en el texto. En ese sentido es una sección muy valiosa para el tribunal ya que puede ver rápidamente de qué fuentes ha bebido el estudiante y valora rápidamente si está basándose en investigaciones potentes (referencias de papers científicos) o si se basa en artículos divulgativos o literatura gris (vídeos, webs, etc). Eso no significa que un buen vídeo no pueda ser una buena referencia, pero el exceso de referencias de este tipo y la carencia del otro puede indicar que se ha profundizado poco.

Tampoco hay un mínimo de citas para un TFG, pero un número excesivamente corto de ellas suele penalizarse. Para que todo sea óptimo, como mínimo debería haber 2 páginas llenas de referencias de calidad. Tampoco hay que pasarse y generar un listado artificialmente grande.

# Apéndice I

Los apéndices son secciones que podrían omitirse sin perder sentido el documento final, pero que al añadirlas agregan cierto valor. Puedes añadir tantos apéndices como necesites, recordando que deben ser unos independientes de otros.

Si por ejemplo vas a añadir un listado de código a un apéndice porque era demasiado largo para ponerlo en su sección correspondiente, no copies y pegues simplemente el listado en una apéndice, añade antes un poco de explicación de qué es y de dónde viene ese trozo de código que vas a copiar (intentando no repetirte respecto a lo dicho en otras partes del TFG). ¡Y ponle una referencia! Es un error común: si añades un apéndice, en algún lugar de la memoria del TFG anterior tienes que nombrarlo. Es decir, no deben existir apéndices que no hayan sido previamente nombrados. En el caso del trozo de código, en algún lugar anteriormente deberías haber escrito “el código completo puede verse en el apéndice X”.

# Acrónimos

Se suele añadir al final del todo, después de los apéndices, una sección de “Acrónimos” en la que se explican los acrónimos usados a modo de referencia.