UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Agronómica y del Medio Natural

TRABAJO FINAL DE GRADO EN BIOTECNOLOGÍA

Método de Identificación de Variaciones Genéticas Relevantes Asociadas a la Distrofia Muscular de Duchenne y de Becker basado en la Inteligencia Artificial Explicable

Universitat Politècnica de València

Curso Académico 2020 – 2021

Valencia, julio de 2021

VRAIN Valencian Research Institute for Artificial Intelligence

PROS Centro de Investigación en Métodos de Producción de Software

Instituto de Investigación Sanitaria La Fe

AUTORA: Ester María López García

TUTOR: Dr. Joaquín Cañizares Sales

COTUTORES: Dr. Óscar Pastor López
Dr. José Fabián Reyes Román
Dr. José María Millán Salvador

RESUMEN

El presente proyecto de fin de grado, impulsado por el grupo de genómica del Centro de Investigación en Métodos de Producción de Software (PROS) de la UPV, miembro del Instituto Valenciano de Inteligencia Artificial (VRAIN), en colaboración con el Dr. José M. Millán, director del Grupo de Investigación en Biomedicina Molecular, Celular y Genómica del IIS La Fe de Valencia, consistió en la identificación de variaciones relevantes asociadas al desarrollo de las enfermedades de distrofia muscular de Duchenne (DMD) y Becker (BMD). Para la consecución de tal tarea, se empleó un método surgido a partir de la unión de dos métodos preexistentes: el método SILE y el método sobre el cual está basado la Inteligencia Artificial Explicable (IAE) .

A lo largo del desarrollo del método, se alcanzaron una serie de objetivos específicos que fueron surgiendo a medida que avanzaba el proyecto. Dichos objetivos consistieron en evaluar y contribuir a la evolución de la plataforma de gestión de datos genómicos "Oráculo Genómico de Delfos" (OraGenDel), desarrollada por el Centro PROS, así como en elaborar propuestas de mejora de la misma. También se determinó la importancia de la base de datos de LOVD, considerada de referencia por parte del personal investigador en el ámbito de las distrofinopatías (el grupo de enfermedades entre las cuales se encuentran la DMD y la BMD), dado que la plataforma OraGenDel no presentaba los conectores necesarios para analizar la información de la misma. Por último, se llevó a término un análisis del significado biológico que albergaban las variaciones que se asociaban a los pacientes de ambas enfermedades (DMD y BMD) y que habían sido clasificadas como relevantes en los experimentos previos. De estas enfermedades destaca el hecho de que, al estar incluidas dentro del grupo de Enfermedades Raras por su baja prevalencia (1-9/100000), suelen contar con menos medios para su investigación. No obstante, con este trabajo, se contribuyó a profundizar en el conocimiento de las mismas a nivel molecular.

Título: Método de Identificación de Variaciones Genéticas Relevantes Asociadas a la Distrofia Muscular de Duchenne y de Becker basado en la Inteligencia Artificial Explicable

Palabras clave: distrofia muscular de Duchenne (DMD); distrofia muscular de Becker (BMD); variaciones genéticas; inteligencia artificial explicable; bases de datos genómicas.

Autor/a del TFG: Dña. Ester María López García

Localidad y fecha: Valencia, julio de 2020

Tutor académico: Dr. Joaquín Cañizares Sales

Cotutor: Dr. Óscar Pastor López

Cotutores colaboradores: Dr. José Fabián Reyes Román | Dr. José María Millán Salvador

RESUM

El present projecte de fi de grau, impulsat pel grup de genòmica del Centre d'Investigació en Métodes de Producció de Software (PROS) de la UPV, membre de l'Institut Valencià d'Intel·ligència Artificial (VRAIN), en col·laboració amb el Dr. José M. Millán, director del Grup d'Investigació en Biomedicina Mol·lecular, Celular i Genómica del IIS La Fe de València, va consistir en l'identificació de variacions rellevants associades al desenvolupament de les malalties distrofia muscular de Duchenne (DMD) i distrofia muscular de Becker (BMD). Per a la consecució de tal tasca, s'emprà un mètode sorgit arran de la unió de dos mètodes preexistents: el mètode SILE i un sobre el què es basa la Intel·ligència Artificial Explicable.

Durant del desenvolupament del mètode, s'alcançaren una sèrie d'objectius específics que van anar apareixent a mesura que avançava el projecte. Tals objectius consistiren, en primer lloc, en evaluar i contribuir a l'evolució de la plataforma de gestió de dades genòmiques "Oracle Genòmic de Delfos" (OraGenDel), desenvolupada pel PROS, així com a elaborar propostes de millora d'aquesta. Un altre objectiu que s'alcançà fou la determinación de l'importància de la base de dades de LOVD, considerada de referència per part del personal investigador a l'àmbit de les distrofinopaties (el grup de malalties dins les quals es troben la DMD i la BMD) i que havien sigut classificades com a rellevants als experiments previs. D'aquestes malaties destaca el fet què, en estar incloses dins del grup de Malalties Rares per la seua baixa prevalència (1-9/100000), solen contar amb menys mitjans per la seua recerca. Malgrat això, amb aquest treball, es va contribuir a aprofundir en el coneixement a nivell mol·lecular d'aquestes malalties.

Títol: Mètode d'identificació de variacions genètiques rellevants associades a la Distrófia Muscular de Duchenne i de Becker basat en l'Intel·ligència Artificial Explicable.

Paraules clau: distrófia muscular de Duchenne (DMD); distrófia muscular de Becker (BMD); variacions genètiques; intel·ligència artificial explicable; bases de dades genòmiques.

Autor/a del TFG: Na Ester María López García

Localitat i data: Valencia, Juliol de 2020

Tutor acadèmic: Dr. Joaquín Cañizares Sales

Cotutor: Dr. Óscar Pastor López

Cotutors col·laboradors: Dr. José Fabián Reyes Román | Dr. José María Millán Salvador

ABSTRACT

The present end-of-degree project, promoted by the genomics group of the Center for Research on Software Production Methods (PROS) of the UPV, member of the Valencian Institute of Artificial Intelligence (VRAIN), in collaboration with José M. Millán, PhD, director of the Molecular, Cellular and Genomic Biomedicine Research Group of the IIS La Fe of Valencia, consisted in the identification of relevant variations associated with the development of Duchenne muscular dystrophy (DMD) and Becker muscular dystrophy (BMD) diseases. In order to achieve this task, a method was used that arose from the union of two pre-existing methods: the SILE method and a method in which Explainable Artificial Intelligence is based.

Throughout the development of this method, a series of specific objectives were achieved, which emerged as the project progressed. These objectives consisted, firstly, in evaluating and contributing to the evolution of the "Delphos Genomic Oracle" genomic data management platform (OraGenDel), developed by PROS, as well as in elaborating proposals for its improvement. Another objective that was achieved was the determination of the importance of the LOVD database, considered as a reference by research staff in the field of dystrophinopathies (the group of diseases among which DMD and BMD are found), given that the OraGenDel platform did not present the needed connectors to analyze the information therein. Finally, an analysis of the biological significance of the variations that were associated with patients with both diseases (DMD and BMD) and that had been classified as relevant in previous experiments was carried out. It is key to point out that given that these diseases are included in Rare Diseases group due to their low prevalence (1-9/100000), they usually have fewer means for their research stands out. However, this work contributed to deepen the knowledge of these diseases at the molecular level.

Title: Method for identification of relevant genetic variations associated with Duchenne and Becker Muscular Dystrophy based on Explainable Artificial Intelligence.

Key words: Duchenne muscular dystrophy (DMD); Becker muscular dystrophy (BMD); genetic variants; explainable artificial intelligence; genomic databases.

Final degree project author: Mrs. Ester María López García

Location and date: Valencia, july 2020

Academic tutor: Joaquín Cañizares Sales, PhD

Cotutor: Óscar Pastor López, PhD

Collaborating cotutors: José Fabián Reyes Román, PhD | José María Millán Salvador, PhD

AGRADECIMIENTOS

Hay veces en las que, en la vida, se suceden una serie de acontecimientos y sincronías que te llevan, sin haberlo planeado, al momento y lugar en que te encuentras. La idea de realizar este TFG surgió de un modo similar. Es verdad que, desde que entré en la carrera de Biotecnología, mi motivación ya iba enfocada, en parte, hacia el tema que trato en este trabajo: avanzar hacia la cura de la enfermedad de Duchenne. No obstante, llegar a colaborar con el Grupo PROS-VRAIN del Dpto. de Sistemas Informáticos y Computación de la UPV, así como con el IIS La Fe de Valencia para materializar lo que antes era simplemente una idea poco definida, no habría sido posible de no ser por esas casualidades de las que hablaba al principio.

Una de estas sincronicidades fue conocer, estando en una conferencia de Lluís Montoliu en la ETSINF (UPV), a Óscar Pastor, quien se convertiría tiempo después en mi cotutor para este trabajo. Aprovecho para agradecerte todo el apoyo y la confianza que has depositado en mí desde el primer momento, por el entusiasmo que te caracteriza y por alentarme a seguir adelante con mis ideas.

Continuando con esta línea, quiero agradecer a José Reyes su inestimable apoyo durante el transcurso de este trabajo. Todas las horas invertidas en ayudarme a encaminar el TFG, las reuniones exprés por Teams, las correcciones de la memoria (aun teniendo el tiempo justo para atender tus muchos otros proyectos)… por todo esto, te mereces mi más sincera gratitud.

Y no quisiera pasar por alto al resto de personas que conforman el maravilloso equipo que da vida al Centro de investigación PROS. Gracias Ana, Alberto y Mireia por acogerme, ayudarme en ciertos momentos del trabajo y hacerme sentir tan cómoda durante estos meses.

Paso ahora a hacer mención al personal del IIS La Fe de Valencia que ha colaborado conmigo para materializar la consecución de este proyecto, en especial a Chema Millán. Gracias por tu disposición, tus consejos y tu cercanía. Y también agradecer a Gema García su apoyo y colaboración. Ambos habéis arrojado luz sobre el camino gracias a vuestros conocimientos en la materia.

Cabe mencionar también a Ximo Cañizares, mi tutor, por haberme acogido como alumna para tutorizar este proyecto. Fuiste tú quien me hizo darme cuenta de mi pasión por la bioinformática. Al igual que también lo hizo la asignatura de AMDB.

Dicho todo esto, dejo atrás el ámbito académico para centrarme en el personal. Continuando, como no podía ser de otra manera, con las sincronicidades o casualidades en relación con este TFG, he de agradecerle a mi amigo Albert el haber tenido aquella conversación el verano pasado, en la cual tuve claro que me quería decantar por este proyecto. Me animaste a creer en mí.

En este sentido, tengo que agradecer también a todas las personas con las que he compartido alegrías, estrés, lloros, abrazos y experiencias enriquecedoras durante mi paso por la universidad, haciendo especial mención al grupo de Cromátidas (Saritxu, Adri, Alejandro, Mariana, David, Yols, Maites, Esther, Edurne, Meri, Dani, Andrea, Mar de Miramar y Alma). Estos cuatro años han sido maravillosos a vuestro lado y confío en que los siguientes lo sigan siendo por igual.

Ahora, esperando no romper demasiado el hilo del discurso, me paso al valenciano porque es la manera más sincera que tengo de referirme a vosotros:

Mónica (Monikín), eres un dels millors regals que m'emporte de la carrera. Malgrat que el teu pas fóra més curt, sempre has estat ahí en els moments decisius. A tú he d'agrair-te tots els consells plens de saviesa i el fet de compartir-me la teua actitut front a la vida.

Alejandro, tú eres un altre dels millors regals que m'emporte de la carrera. Tinc moltíssima sort d'haver-te conegut. A tú t'agraïsc totes les converses, els consells, els aprenentatges i les vivències, així com tot el recolçament rebut durant tot aquest temps.

Cèsar, gràcies per estar ahí en les bones i en les no tan bones, per les converses esclaridores sobre el TFG i sobre la vida, per les correccions, per les experiències viscudes i per ser un dels meus punts de referència.

Retomo el castellano para referirme a ti, Sandra (Xicuela), a quien he tenido muy presente en mi vida en el último año, sobre todo. Gracias por ser esa amiga en la que poder confiar, ese apoyo en los momentos de debilidad y esa puerta de escape de la rutina. A ti te agradezco también haberme abierto los oídos al canto de las aves.

Puedo decir orgullosa que he tenido y tengo muchísima suerte de estar rodeada de personas tan extraordinarias como vosotras, que me han arropado en todo momento y me han animado a seguir adelante instándome a confiar en mí misma. Pero no sólo han sido personas quienes me han mostrado su apoyo, pues no puedo dejar pasar por alto a mi gato, Pipo. A él le agradezco su compañía en las largas noches en vela estudiando, a raíz de lo cual puedo afirmar que se ha sacado la carrera conmigo.

Y me dejo para el final una de las piezas clave de mi vida, mi familia. Qué decir de vosotros que no sepáis ya. A mis padres, tengo que agradeceros todo, desde haber podido llevar a cabo este trabajo final de grado hasta haber hecho que me convierta en la persona que soy ahora. Habéis sido mi apoyo incondicional durante todos estos años, mi refugio, la fuente de sabiduría, los responsables de ofrecerme todas las oportunidades de formación y aprendizaje que estaban a vuestro alcance, entre muchas otras cosas y, a pesar de todo ello, no tengo palabras para expresar lo verdaderamente admirables que sois y lo profundamente agradecida que estoy. Os quiero.

Quiero hacer mención también a mis abuelos, pues pese a que algunos ya no estén aquí, sé que les habría hecho especial ilusión verme finalizar el grado.

Y, por último, a mi hermano. Es a ti, Miguel, a quien va dedicado este trabajo. Has sido mi verdadera motivación para embarcarme en un proyecto como este. He de admitir que no ha sido fácil tratar de gestionar algo que toca tan de cerca, pero saber que estaba colaborando en el empeño por comprender mejor la enfermedad de Duchenne me daba fuerzas para seguir adelante. Te doy las gracias por ser el mejor hermano que podía tener, por ser un punto clave en mi vida y por el continuo aprendizaje. Por mucho que discutamos, y haya ocasiones en que con razón ;), siempre has de saber que voy a estar al pie del cañón para lo que necesites. Te quiero.

ÍNDICE TEMÁTICO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

LISTA DE ACRÓNIMOS Y SIGLAS

AAV	*Adeno-associated virus*
ABD	*Actin Binding Domain*
ACMG	*American College for Medical Genetics and Genomics*
ADP	*Adenosine diphosphate*
AMP	*Association for Molecular Pathology*
BBDD	Bases de Datos
BLASTX	*Basic Local Alignment Search Tool X*
BMD	*Becker muscular dystrophy*
cDNA	*complementary Deoxyribonucleic Acid*
CK	*Creatine Kinase*
CRISPR/Cas9	Cluster Regularly Interspaced Short Palindromic Repeats / CRISPR associated protein 9
CSV	*Comma-Separated Values*
DAPC	*Dystrophin-associated protein complex*
DGVa	*Database of Genomic Variants archive*
DMD	*Duchenne muscular dystrophy*
DNA	*Deoxyribonucleic Acid*
EMA	*European Medicines Agency*
EMBL-EBI	*European Molecular Biology Laboratory - European Bioinformatics Institute*
EVA	*European Variation Archive*
FDA	*Food and Drug Administration*
GWAS	*Genome-Wide Association Studies*
HGVS	*Human Genome Variation Society*
IA	Inteligencia Artificial
IAE	Inteligencia Artificial Explicable

LOVD	*Leiden Open Variations Database*
LTBP4	*Latent-transforming growth factor beta-binding protein 4*
MC	Modelado Conceptual
MCGH	Modelo Conceptual del Genoma Humano
MEC	Matriz Extracelular
MIM	*Mendelian Inheritance in Man*
ML	*Machine Learning*
MLPA	*Multiplex Ligation Probe Amplification*
MP	Medicina de Precisión
mRNA	m*essenger* RNA
NCBI	*National Center for Biotechnology Information*
ncRNA	*non-coding* RNA
NGS	*Next Generation Sequencing*
NHGRI	*National Human Genome Research Institute*
nNOS	*Neuronal Nitric Oxide Synthase*
NO	Óxido Nítrico
O_2	Oxígeno
OraGenDel	Oráculo Genómico de Delfos
pbs	Pares de bases
PCR	*Polymerase Chain Reaction*
PolyPhen2	*Polymosrphism Phenotyping v2*
PROS	*Research Center on Software Production Methods*
RNA	*Ribonucleic Acid*
SI	Sistemas de Información
SIFT	*Sorting Intolerant from Tolerant*
SILE	*Search – Identification – Load – Exploitation*
SNV	*Single Nucleotide Variant*
TSV	*Tab-Separated Values*

URL	*Uniform Resource Locator*
VRAIN	*Valencian Research Institute for Artificial Intelligence*
XLDC	*X-linked dilated cardiomyopathy*
XLSX	*Excel Microsoft Office Open XML Format Spreadsheet file*

1. INTRODUCCIÓN

En la actualidad, la gran cantidad de datos genómicos generados a partir de las tecnologías de secuenciación de alto rendimiento o de *Next Generation Sequencing* (NGS) ha dado lugar a la aparición de un amplio número de repositorios de datos genómicos que, lejos de ofrecer datos entendibles, contribuye a la dificultad de encontrar información de calidad de la que poder extraer conocimiento de valor. Asimismo, estas tecnologías de secuenciación masiva emplean, a su vez, métodos distintos de tratamiento de la información, los cuales dan lugar a diversas formas de referirse a un mismo elemento. La problemática derivada de esto último ha potenciado el desarrollo de la bioinformática. Esta nueva disciplina, que aúna el conocimiento derivado de la biotecnología y de la informática, se propone varios objetivos (Solanki *et al.*, 2020). En este trabajo de investigación, no obstante, se persigue uno en concreto, que consiste en facilitar el tratamiento y gestión de datos para extraer y explotar información útil y valiosa.

En el presente trabajo, se pretende determinar qué variaciones están mayormente implicadas en el desarrollo de ciertos fenotipos patogénicos, específicamente en los casos de la distrofia muscular de Duchenne (DMD; MIM 310200) y de la distrofia muscular de Becker (BMD; MIM 300376) (OMIM, s.f). Asimismo, mediante la consecución de los objetivos específicos planteados en el **Apartado 2**, se busca contribuir a un mayor conocimiento de estas enfermedades, con el fin de mejorar la calidad en aspectos de prevención y diagnóstico de los pacientes. Finalmente, se plantea colaborar con los esfuerzos generalizados por parte de la comunidad científica por alcanzar un mayor entendimiento de los datos genómicos, así como una obtención y gestión eficiente de los mismos.

Más concretamente, en este trabajo se desarrolla un método de identificación de variaciones relevantes que puedan estar asociadas a los fenotipos de estudio u otros fenotipos. Del mismo modo, se valida también el funcionamiento de la plataforma *software* denominada "Oráculo Genómico de Delfos" (OraGenDel), la cual consiste en un sistema de extracción, clasificación y análisis de datos genómicos, desarrollado por el grupo del Centro PROS, miembro del Instituto Valenciano de Inteligencia Artificial (VRAIN), de la Universitat Politècnica de València, cuyo objetivo es tratar de abordar los problemas que presenta el actual "caos" de datos genómicos mediante la creación de un repositorio universal capaz de albergar información genómica con la finalidad de disponerla al alcance de todo profesional del ámbito clínico que la requiera. Con esto último, se pretende facilitar las tareas de los expertos en los procesos de toma de decisiones.

Para realizar esta validación, se llevan a cabo una serie de pasos que constituyen los fundamentos del Método SILE (explicado en el **Apartado 1.1.2**), elaborado por Ana León en su tesis doctoral (2019). Este método, visto a grandes rasgos, consiste en la búsqueda de repositorios genómicos para la extracción de la información; la identificación de las variaciones extraídas siguiendo un riguroso algoritmo de clasificación; la carga en el repositorio de OraGenDel de los datos curados y respaldados por un soporte científico de valor; y, por último, en la explotación de los datos obtenidos mediante el análisis de las características que presenta el conjunto de variaciones. Ahora bien, el seguimiento de estas fases se ha enmarcado en este trabajo, de manera novedosa y original, dentro del contexto de la Inteligencia Artificial Explicable (IAE), debido a las ventajas que esto le confiere y que se explican en detalle en el siguiente apartado (**1.1**). Por esta razón, el método empleado para la consecución de los objetivos

propuestos consta, en definitiva, de cinco etapas, que se desarrollan en el apartado de Método (**Apartado 4**).

Además, un hecho relevante que caracteriza a este trabajo de investigación es la colaboración que se establece con el personal investigador en DMD y BMD. Dicho personal, pertenece al ámbito clínico y de análisis genético del Grupo de Investigación en Biomedicina Molecular, Celular y Genómica del IIS La Fe de Valencia, dirigido por el Dr. José M. Millán, considerado un experto en el campo de las distrofinopatías. Por esta razón, las recomendaciones realizadas por los mismos en cuanto al desarrollo del proyecto, se incorporan al planteamiento de los objetivos específicos (**Apartado 2**). En concreto, el tercer (OBJ.3) y el cuarto objetivo (OBJ.4), los cuales van enfocados a determinar la utilidad de una base de datos (BBDD), LOVD, la cual no se encuentra disponible en el OraGenDel, y a estudiar las diferencias entre los fenotipos de DMD y BMD, respectivamente, son establecidos en base a dichas recomendaciones.

A continuación, antes de explicar punto por punto el procedimiento que se ha seguido en el desarrollo de este trabajo final de grado, se introducirán los conceptos necesarios para enmarcar el contexto del trabajo. Dichos conceptos son: la Inteligencia Artificial Explicable y las Distrofinopatías.

1.1. INTELIGENCIA ARTIFICIAL EXPLICABLE (IAE)

El término Inteligencia Artificial (IA) hace alusión a la tendencia tecnológica que cubre cualquier nuevo desarrollo capaz de automatizar y optimizar tareas que, tradicionalmente, requerían de la inteligencia humana. Desde los inicios de la ciencia de la computación, uno de los pilares básicos de la misma ha sido su incentivo por facilitar la vida de los seres humanos mediante la automatización de una gran diversidad de procesos. No obstante, nunca antes su implicación había alcanzado niveles tan elevados de decisión como hasta la fecha (p.ej., medicina de precisión, conducción autónoma, etc.), ya que el notable avance que ha experimentado en los últimos años la IA, ha guiado a la misma al desempeño de un papel cada vez más relevante en los sistemas de toma de decisión y en los procesos autónomos (Kulkarni *et al.* 2020; Spreeuwenberg, 2019).

Sin embargo, la IA no está exenta de problemas. La mayor barrera que presenta es la de la "*explicabilidad*", es decir, la incapacidad de argumentar qué decisiones rigen los procesos que dan lugar a los resultados obtenidos (Barredo *et al*., 2020; Tjoa y Guan, 2015). Afortunadamente, la detección del problema ha permitido desarrollar estrategias para abordarlo, las cuales constituyen los ejes que articulan el concepto de IAE.

La IAE surge con el propósito de incrementar la robustez, el entendimiento y la transparencia de los sistemas de IA. En la actualidad, el área de la salud está experimentando una tendencia al alza en el campo de la Medicina de Precisión (MP) (Hulsen *et al.*, 2019), la cual viene impulsada por tecnologías emergentes como la IA o el análisis de grandes cantidades de datos mediante el empleo de herramientas de aprendizaje automático (Ho *et al.* 2020). En este contexto, es clave promover el empleo de la IAE para asegurar una toma de decisiones basada en el conocimiento absoluto de todas las etapas del proceso.

En el ámbito de la gestión de datos genómicos, esto se traduce en una serie de argumentos lógicos que justifican las decisiones ejercidas por los sistemas *software*, los cuales le confieren al profesional clínico la información necesaria para aceptar o rechazar

las conclusiones generadas por el programa informático. De acuerdo con lo expuesto en el libro de Spreeuwenberg "AIX: AI Needs eXplanation" (2019), un sistema que pretende ser útil a la hora de justificar la toma de decisiones en el ámbito clínico, ha de ser capaz de:

i) Indicar las características principales que conducen a los resultados obtenidos (p.ej., cuando se realiza el filtrado de un fichero con datos crudos – no curados – y se genera un fichero de salida distinto del fichero de entrada, se han de especificar los criterios utilizados para el filtrado).
ii) Proveer información acerca del contexto en el cual opera el sistema (p.ej., explicando previamente el dominio de trabajo, como se muestra en el **Apartado 4.1**).
iii) Informar de las situaciones para las cuales el sistema podría presentar controversias (p.ej., desarrollando estrategias para el análisis, como en el **Apartado 4.4**).
iv) Permitir el seguimiento del flujo de trabajo (p.ej., mediante la representación del proceso que sigue el programa haciendo uso del modelado conceptual y los diagramas de flujo).

Una vez vista la importancia de la IAE y la justificada necesidad de su aplicación, se procede a profundizar en la misma mediante la explicación del método de IAE desarrollado por Silvie Spreeuwenberg (**Subapartado 1.1.1**). Este método, junto con el método SILE que se explicará en el subapartado posterior (**1.1.2**), da lugar al método que se emplea en el presente trabajo (**Apartado 4**).

1.1.1. Método de IAE

Las ventajas de la IAE que explica Spreeuwenberg en su libro (2019) y que han sido expuestas en la introducción de este apartado, son prometedoras en la teoría, pero en ausencia de una manera clara de aplicarlas, difícilmente pueden probarse sus beneficios. Por consiguiente, a continuación, se muestra el método elaborado por la experta en IA Silvie Spreeuwenberg, el cual define los pasos que se deben proseguir si se pretende incorporar las ventajas que confiere la IAE.

1. **Comprender el dominio**. Los conceptos del dominio de trabajo han de estar definidos de manera clara y unívoca. Con este fin, suele ser de utilidad el empleo de mapas conceptuales que representan de manera gráfica el contexto de la tarea que se pretende llevar a término.

2. **Determinar la tarea de decisión**. Los ámbitos que puede abarcar la aplicación de la IA son prácticamente ilimitados, por lo que es determinante acotar el área de actuación y establecer el objetivo que se pretende alcanzar. Se suele recurrir al empleo de modelos de decisión o algoritmos con tal de facilitar la comprensión.

3. **Seleccionar los datos adecuados y mejorar su calidad.** Se trata de uno de los pasos más importantes, pues el resultado que se obtenga dependerá en gran medida de la fuente de datos a partir de la cual se recoja la información en la que se vaya a basar el algoritmo de IA.

4. **Establecer la técnica de IA capaz de proporcionar resultados.** En base a las características de las tareas y los datos, resultado de los análisis anteriores, se debe determinar cuál es la mejor solución para llevar a cabo la tarea planteada.

5. **Generar buenas explicaciones.** Esta etapa es la más relevante del método de IAE, pues refleja la culminación de los esfuerzos realizados en cada una de las etapas anteriores para dotar de consistencia y fiabilidad a los resultados obtenidos. Además de la confianza que esto genera en el modelo, las explicaciones ayudan a encontrar problemas en los datos, como pueden ser los resultados anómalos o aquellos que no se corresponden con las recomendaciones de los expertos. Esto último, al haber sido obtenido de manera clara y transparente, contribuye a la mejora y refinamiento del algoritmo, lo cual forma parte del siguiente paso.
6. **Evaluar las soluciones con el transcurso del tiempo.** Una vez el sistema se pone en funcionamiento, se debe analizar con objeto de optimizar los resultados. Para ello, se emplea un bucle de retroalimentación en el cual el personal implicado toma decisiones para la mejora y evolución del sistema, siempre tomando como punto de partida las explicaciones generadas.

En el actual proyecto de fin de grado se emplea este método como base estructural para albergar al Método SILE, de identificación de variaciones genéticas relevantes, el cual se explicará a continuación.

1.1.2. Método SILE

El Método SILE, cuyas siglas hacen referencia a cada una de las etapas que lo conforman: *Search* (Búsqueda), *Identification* (Identificación), *Load* (Carga) y *Exploitation* (Explotación), se trata de un abordaje sistemático para la gestión eficiente de los datos genómicos. En cuanto a su origen, la tesis doctoral de A. León (2019) establece las bases sobre las que se fundamenta este método. Sin embargo, han sido varias las publicaciones científicas que han surgido como consecuencia de su aplicación y en las cuales han ido evolucionando ciertos aspectos del mismo (Costa *et al*. 2020; León Palacio *et al*. 2018). Este hecho es muestra del dinamismo que presenta y de la capacidad de adaptarse a las fluctuaciones del entorno de trabajo, a parte de haber permitido validar su utilidad con el estudio de fenotipos concretos como la migraña, la epilepsia o la fibrosis quística, entre otros.

La razón por la que surge este método es dar solución a la problemática del caos de datos que deriva de la aparición de las NGS. La mayoría de los datos genómicos generados por estas tecnologías son incluidos en BBDD que han sido desarrolladas con objeto de albergar un tipo de información en concreto, pero que carecen de la interrelación necesaria para conferir la visión sistémica que requiere un campo tan vasto como el de la información genómica. Esto desemboca, en última instancia, en inconsistencias, redundancias y dispersión en lo que concierne a los datos, así como en diferentes representaciones de un mismo concepto, que contribuyen a aumentar el ruido y a incrementar la variabilidad en su calidad (Palacio y Pastor, 2018).

El Método SILE pretende automatizar la búsqueda de variaciones asociadas a un fenotipo concreto y, para ello, se siguen las siguientes etapas:

1. ***Search***. Consiste en la búsqueda y selección de las fuentes de datos adecuadas a partir de las cuales extraer la información requerida según el estudio a realizar.

2. ***Identification***. Se basa en la identificación de las formas de acceso pertinentes a los repositorios, así como de la información que se pretende analizar. En esta etapa, se efectúa también una transformación de la información mediante el filtrado y la unificación del formato que presentan los datos. Por último, sobre la información ya procesada, se realiza un control de calidad de los datos, el cual determina qué variaciones son relevantes clínicamente.

3. ***Load***. Radica en almacenar la información de calidad extraída, procesada y seleccionada, habiendo previamente considerado el uso que se le va a dar a la misma.

4. ***Exploitation***. Se basa en la explotación de la información para extraer el conocimiento.

La aplicación de este método en el presente trabajo permite disponer de un ambiente de trabajo para la búsqueda de variaciones asociadas a la DMD y la BMD en las fuentes de datos pertinentes (**Subapartado 3.2**). También, se consigue la identificación en dichas fuentes de las variaciones que se consideran relevantes desde el punto de vista clínico. Y, por último, facilita la posterior carga y explotación de las variaciones en la plataforma OraGenDel, la cual se explica en el apartado de Materiales (**Apartado 3**).

Con la explicación del Método SILE se pone fin al subapartado Inteligencia Artificial Explicable (**1.1**), donde se ha mostrado el contexto de trabajo más relacionado con la ciencia de datos y los sistemas *software*. En el siguiente subapartado, se expondrá el otro eje que vertebra el proyecto, que es el fenotipo sobre el cual se va trabajar. Los fenotipos de estudio son la distrofia muscular de Duchenne y la distrofia muscular de Becker, los cuales se agrupan bajo el concepto de Distrofinopatías.

1.2. DISTROFINOPATÍAS

Las distrofinopatías constituyen un grupo de desórdenes hereditarios ligados al cromosoma X, cuya principal afectación es el tejido muscular esquelético y el cardíaco, los cuales se degradan de manera progresiva con el transcurso del tiempo como consecuencia de un mal funcionamiento de la proteína distrofina (UniProtKB/Swiss-Prot "P11532") o de la ausencia de la misma. Esta anomalía suele venir dada por alteraciones genéticas producidas en el gen *DMD*, que codifica para esta proteína.

El gen *DMD* (RefSeqGene "LGR_199"; NCBI Reference Sequence "NG_012232.1"), localizado en el brazo corto del cromosoma X (Xp21.2) y cuya extensión es de 2.2 Mb, se trata del gen más largo del genoma humano. En términos de filogenia, presenta dominios altamente conservados entre especies, principalmente en el extremo 3' (Yotova *et al*., 2007). En humanos, consta de 79 exones, que codifican para distintas isoformas. Las más largas son la del músculo ("Dp427m"), el córtex cerebral ("Dp427c") y las células cardiacas de Purkinje ("Dp427p") . El resto de isoformas presentan un menor tamaño y se expresan en menor medida en retina ("Dp260"), nervio periférico ("Dp116"), riñones ("Dp140") o de forma ubicua ("Dp71") (Chelly *et al*., 1990; Górecki *et al*., 1992; D'Souza *et al*., 1995; Lidov *et al*., 1995; Hugnot *et al*., 1992).

Los principales fenotipos asociados a alteraciones genéticas patogénicas de este gen son, de mayor a menor severidad, la distrofia muscular de Duchenne y la distrofia

muscular de Becker. Otras distrofinopatías que constituyen formas intermedias son la cardiomiopatía dilatada ligada a X (XLDC; MIM 302045), el síndrome de mialgia y calambres o la presentación de forma aislada de niveles elevados de creatina quinasa (CK) en suero. Además, en ocasiones, las mujeres portadoras pueden ser sintomáticas, presentado también cuadros de gravedad variable (Vieitez *et al*., 2016; Fratter *et al*., 2020). Cabe destacar también que, pese a que estas enfermedades presentan un claro patrón de herencia recesiva, se ha observado que un tercio del total de afectados por DMD y BMD lo están como consecuencia de la aparición de mutaciones *de novo* (García *et al*., 2014; Caskey *et al*.,1980).

La consecuencia directa de las alteraciones en el gen *DMD* son problemas en el funcionamiento de la distrofina, como bien refleja el nombre "distrofinopatía", el cual etimológicamente está referido a la enfermedad o daño ("-patía") de la distrofina ("distrofin-"). Esta última, se trata de una proteína citoplasmática de 427 kDa cuya función principal consiste en el anclaje de la matriz extracelular (MEC) al citoesqueleto por medio de la F-actina y cuya estructura se muestra en la **Figura 1**. Esta unión permite disipar la fuerza contráctil producida por el movimiento de las fibras musculares evitando, de este modo, el deterioro del sarcolema. Un hecho interesante es que, en el músculo esquelético sano, esta proteína únicamente supone un 0.002% del total de proteína muscular, pero su ausencia conduce a un marcado detrimento de la funcionalidad del músculo (Hoffman *et al*., 1987).

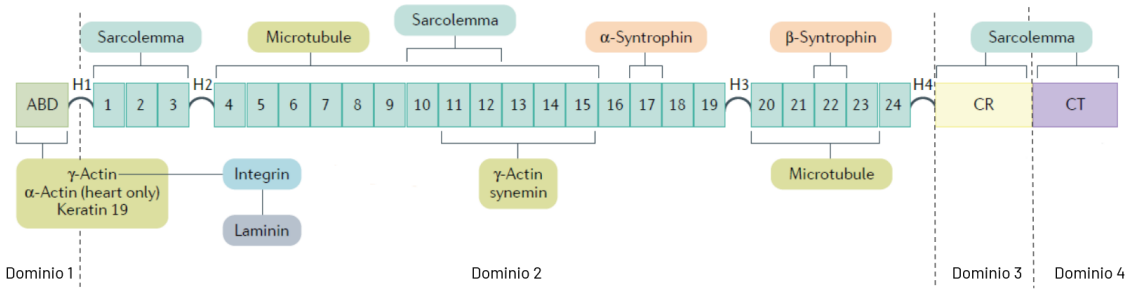

Figura 1. Estructura lineal de la proteína distrofina. Fuente: imagen adaptada de Duan et al., 2021.

La estructura de la distrofina consta de cuatro dominios principales: i) un dominio de unión a actina (ABD), ii) un rodo dominio central, iii) un dominio rico en cisteína y iv) el extremo carboxilo terminal. El dominio ABD incluye dos dominios homólogos a la calponina (CH1 y CH2), los cuales se encargan de la unión a la F-actina (Way *et al*., 1992). El segundo, el rodo dominio central, contiene 24 repeticiones de espectrina, las cuales presentan estructura de tipo triple α-hélice y un segundo dominio de unión a actina entre la repetición R11 y R15. Además, también presenta zonas de unión al sarcolema (R10-R12), a microtúbulos (R4-R15, R20-R23) y a algunas subunidades de sintrofina (R17, R22). Es relevante destacar que las repeticiones de espectrina no son continuas completamente, pues presentan 4 espaciadores (H) ricos en prolina, el último de los cuales contiene además un dominio triptófano-triptófano o WW de interacción con β-distroglicano) que confieren elasticidad a la distrofina. Con respecto al tercer dominio, el dominio rico en cisteína, su función radica en fijar la unión de la proteína al sarcolema, mediante la interacción con la anquirina-B, una proteína adaptadora (Ayalon *et al*., 2008). Y, finalmente, el dominio C-terminal contiene dos polipéptidos de estructura helicoide-enrollado similar a la espectrina que interaccionan con el complejo distrobrevina-sintrofinas-nNOS de señalización.

El funcionamiento de la distrofina está incluido dentro de un complejo en el que intervienen proteínas asociadas de tipo señalizadoras, formadoras de canales y estructurales. Este complejo se denomina DAPC (por sus siglas en inglés "*Dystrophin-associated protein complex"*) (Gao y McNally, 2015) y se muestra en la **Figura 2**.

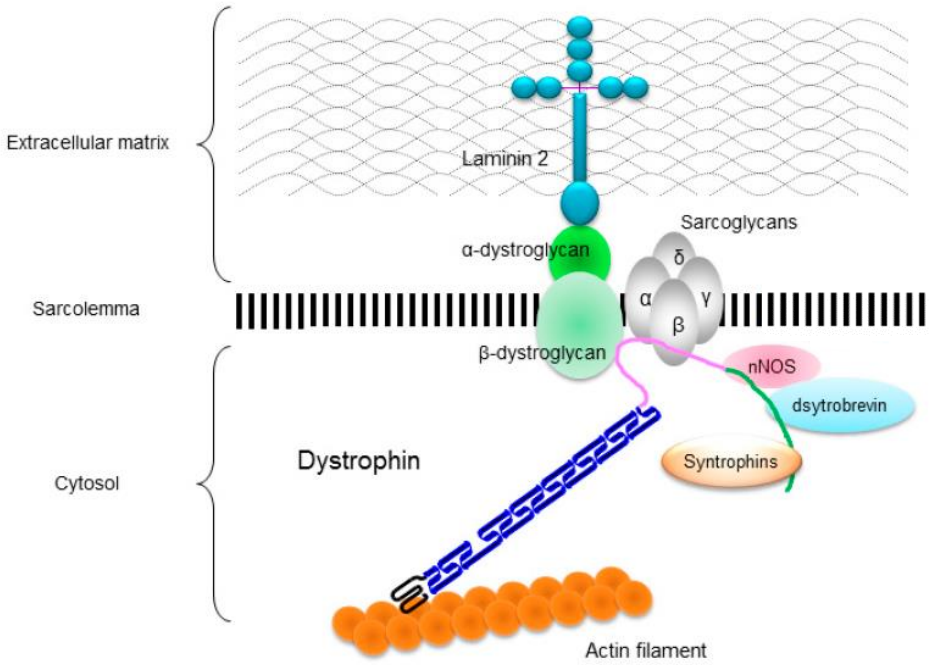

Figura 2. Complejo DAPC. Fuente: Imagen adaptada de Nakamura, 2015.

El complejo DAPC se divide en tres secciones, en base a la localización celular de los elementos que lo conforman: i) parte extracelular (α-distroglicano), ii) parte transmembrana (β-distroglicano, sarcoglicanos) y iii) parte citoplasmática (distrofina, distrobrevina, sintrofinas y óxido nítrico sintasa neuronal o nNOS). El α-distroglicano se ubica, como consecuencia de su elevada glicosilación, en la parte extracelular, donde actúa como proteína receptora de laminina-2, estableciendo la unión entre el complejo y la MEC (Ibraghimov-Beskrovnaya *et al*., 1992). En la parte transmembrana o del sarcolema, el β-distroglicano permite la interacción de manera indirecta entre el α-distroglicano y la distrofina. Asimismo, los sarcoglicanos dotan a la unión que forma el dímero de distroglicanos (α y β) de la fuerza necesaria para proveer estabilidad al mismo, así como protección mecánica al sarcolema. Y, por último, en la parte intracelular es donde se localiza la distrofina, la cual se une por el extremo N-terminal a las fibras de actina, permitiendo la interacción entre la lámina basal de la MEC y la parte más interna de la célula. Además, en esta parte también se localizan las sintrofinas, la distrobrevina y el nNOS, los cuales interactúan entre sí para enviar a los vasos sanguíneos, mediante la liberación de NO, la señal de dilatación. Esto posibilita que haya un mayor flujo de sangre que permite transportar una mayor cantidad de O_2 y nutrientes al músculo durante el ejercicio (Gao y McNally, 2015).

Una vez vistos los aspectos moleculares y establecido el contexto en que se desarrollan estas enfermedades, en los siguientes subapartados se profundizará en las características propias de los dos fenotipos principales derivados del fallo del proceso de contracción muscular en el que está implicada la proteína distrofina. Asimismo, también se mostrarán las terapias más novedosas existentes en este campo.

1.2.1. Distrofia Muscular de Duchenne (DMD)

La distrofia muscular de Duchenne o DMD es la manifestación más severa de las distrofinopatías comentadas anteriormente. Fue descrita por primera vez en 1860 por el neurólogo francés Guillaume Duchenne (Tyler, 2003) y, en la actualidad se engloba dentro del grupo de "enfermedades raras" en base a su baja prevalencia (1-9/100000), pese a ser la distrofia muscular más común, con una incidencia de 1 afectado por cada 3500 nacimientos (ORPHANET, 2020).

Su cuadro clínico se caracteriza por el retraso en la edad de inicio de deambulación, signo de Gowers positivo, dificultad para correr y subir escaleras, hiperlordosis, hipertrofia de gastrocnemio y caídas frecuentes. Estos signos se suelen observar a la edad de 3-5 años. Una vez se sospecha de esta enfermedad, en base al cuadro clínico y a la herencia familiar, se procede al análisis bioquímico de los niveles de CK sérica, una enzima citoplasmática que cataliza la fosforilación del ADP tras la contracción muscular, y cuyos niveles se ven aumentados en el suero de los pacientes debido a la ruptura del sarcolema (Łoboda y Dulak, 2020). Asimismo, existen otros tipos de biomarcadores diagnósticos, como los microRNAs (p.ej., miR-31a, miR-206) o, incluso, biomarcadores pronósticos, como algunos factores genómicos (p.ej., la presencia de un haplotipo determinado, consistente en 4 variaciones, que contribuye a un incremento en los niveles de LTBP4, una proteína cuya función se ha observado que promueve el retraso de la edad de pérdida de ambulación en los pacientes) (Weiss *et al*., 2018; Szigyarto y Spitali, 2018).

A nivel genético, la DMD se asocia principalmente con grandes deleciones (~70%) y con mutaciones puntuales (SNV) que desplazan la pauta de lectura de la maquinaria de traducción, produciendo la aparición de codones de *stop* prematuros (Vieitez *et al*. 2016; Nascimento Osorio *et al*. 2019). Los transcritos mutados son susceptibles de ser eliminados por el mecanismo de degradación del RNA (*Ribonucleic Acid*) mensajero mediado por mutación terminadora (o *nonsense-mediated mRNA decay)*, al igual que ocurre con las proteínas que tienen truncado el extremo C-terminal, las cuales son altamente inestables (Gao y McNally, 2015). El análisis que permite detectar las deleciones y duplicaciones se trata del MLPA, una variación de la PCR que permite amplificar múltiples secuencias empleando un único par de cebadores (Schouten *et al*., 2002). No obstante, en caso de tratarse de una SNV, la técnica MLPA no es lo suficientemente sensible, por lo que se debe realizar una biopsia muscular para analizar los niveles de distrofina mediante imnunohistoquímica (aunque se intenta evitar por su invasividad) y/o una secuenciación de la región genómica donde se localiza el gen *DMD* para encontrar la variación exacta que conduce a este fenotipo (Lim *et al*., 2011; Nascimento Osorio *et al*., 2019).

Pese a que no existe una cura para la DMD, en los últimos años se han sucedido una serie de avances sin precedentes que han revolucionado el campo de las terapias para esta enfermedad. De manera tradicional, se administran corticoides debido a los beneficios a largo plazo sobre la función motora, cardiaca y respiratoria que presentan (Moxley *et al*., 2005). Sin embargo, más recientemente, han sido aprobados otros tratamientos de manera condicional, para mutaciones específicas, que están mostrando resultados prometedores. Algunos de ellos son: un fármaco que permite saltar los codones de *stop* prematuros producidos por mutaciones *nonsense* (Ataluren), aprobado condicional por la EMA (*European Medicines Agency*) en 2014 (Haas *et al*., 2015); y la terapia de salto de exón para los exones 51 (Eteplirsen) y 53 (Golodirsen), aprobada por la FDA (*Food and Drug Administration*) en 2016 y en 2019, respectivamente (Luce *et al*.,

2021). Asimismo, existen otros tratamientos en desarrollo como la terapia génica con CRISPR/Cas9, actualmente en fase de ensayos preclínicos probados en ratones y en perros (Zhang *et al*., 2020; Amoasii *et al*. 2018), o el empleo de vectores virales adeno-asociados (AAV) para introducir un cDNA (*complementary Deoxyribonucleic Acid*) codificante para la microdistrofina, una proteína de menor tamaño que la distrofina pero que contiene los dominios esenciales para interaccionar con el DAPC (Duan, 2018).

Los pacientes con DMD tienen una esperanza de vida limitada, la cual se ha visto, en algunos casos, incrementada hasta los cuarenta años gracias a la mejora en las técnicas de diagnóstico, al desarrollo de las guías clínicas de actuación y a los avances científicos que se han ido llevando a cabo en el campo estos últimos años (Ryder *et al*., 2017). Asimismo, es de destacar la importancia que cobra recibir una atención multidisciplinar por parte de profesionales de todas las áreas de la salud para el manejo de estas patologías, tanto de la DMD como de la BMD, que se explica a continuación.

1.2.2. Distrofia Muscular de Becker (BMD)

La distrofia muscular de Becker o BMD se trata de una distrofinopatía menos grave que la que se ha explicado en el apartado anterior (**1.2.1**). No obstante, existe un amplio espectro para esta enfermedad, que abarca desde manifestaciones subclínicas hasta un fenotipo prácticamente similar al de la DMD (Koenig *et al*., 1989). Su origen se remonta a 1950, cuando fue descrita por primera vez por el médico genetista alemán Peter Emil Becker. La prevalencia en la población es de 2/100000 y la incidencia: 1 afectado por cada 16000 nacimientos (ORPHANET, 2020).

Los signos y síntomas que presentaban los pacientes con DMD a la edad de 3-5 años, en este caso, aparecen sobre los 10-15 años. Estos, además, mantienen la capacidad de ambulación durante varios años más que los pacientes con DMD, lo cual es considerado por muchos ensayos clínicos como un aspecto distintivo para la aceptación de pacientes en estos estudios (Mendell *et al*., 2015).

Pese a que el cuadro clínico puede proporcionar cierta información para realizar un diagnóstico provisional, es imprescindible llevar a cabo un análisis genético para determinar que se trata de una distrofinopatía. Esto se puede hacer con la prueba MLPA sobre el gen *DMD*. Ahora bien, para efectuar un diagnóstico diferencial con respecto a la DMD, se debe proceder bien con un análisis inmunohistoquímico de biopsia muscular, en el que la cantidad de distrofina se verá reducida pero no completamente ausente, o bien con la secuenciación de la región genómica del gen *DMD*, siendo esto último imprescindible para determinar el origen exacto de la alteración (Bello *et al*., 2016).

Al contrario que ocurría en la DMD, los pacientes con BMD suelen presentar mutaciones que no alteran el marco de lectura. En cuanto a la alteración más frecuente, los pacientes con BMD normalmente presentan deleciones, como en DMD, las cuales en este caso no imposibilitan que se traduzca la proteína (Koenig *et al*., 1989). Por otro lado, cuando se trata de mutaciones de un único nucleótido, estas suelen tratarse de las de tipo *missense* en vez de *nonsense* (Gao y McNally, 2015; Vieitez *et al*., 2016).

Con respecto a los tratamientos, se ha observado que la administración de corticoides no proporciona beneficios a los pacientes, al contrario que ocurría con la DMD, por lo que no se recomiendan (Bäckman y Henriksson, 1995). Sin embargo, no se están llevando a cabo demasiados ensayos clínicos específicos para la BMD, pues

generalmente suelen ir destinados a pacientes con DMD y BMD o únicamente con DMD. Algunos estudios, como el realizado en 2015 por Mendell y su equipo (2015), entre otros (Wagner *et al*., 2008), han demostrado el potencial que presenta la inhibición de la ruta de la miostatina, una proteína inhibidora del crecimiento muscular. Otro ejemplo de posible tratamiento está basado en la detección de grandes deleciones que mantienen el marco de lectura. Se ha observado que el fenotipo que se produce en ausencia del exón 5 es más severo que el esperado en la BMD, mientras que la deleción de los exones 3 a 9 produce un fenotipo de BMD prácticamente asintomático (Toh *et al*., 2016; Nakamura *et al*., 2016). Este hecho podría ser de provecho, tanto para pacientes con DMD como BMD, si se consiguen eliminar los exones del 3 al 9 en aquellos pacientes que tengan deleción del exón 5 (Sheikh y Yokota, 2020).

2. OBJETIVOS

El objetivo principal del presente Trabajo de Final de Grado es identificar y obtener variaciones genéticas relevantes para las enfermedades distrofia muscular de Duchenne y distrofia muscular de Becker, a partir de una búsqueda exhaustiva en diversos repositorios de datos genómicos. Dicha identificación de variaciones genéticas relevantes es necesaria debido al gran número de BBDD existentes y a los problemas que derivan del número creciente de información que albergan.

Asimismo, también es relevante distinguir los objetivos específicos, a parte del objetivo principal, con el fin de posibilitar un fácil seguimiento y un mejor entendimiento de los puntos clave del trabajo. Los objetivos específicos son lo que se plantean a continuación:

- **OBJETIVO 1 (OBJ. 1)**
  Enmarcar el método SILE dentro de un método de IAE, lo cual implica la creación de un nuevo método que surge a partir de la fusión de dos preexistentes.

- **OBJETIVO 2 (OBJ. 2)**
  Aplicar el algoritmo de clasificación de variaciones genéticas relevantes asociadas a la DMD y a la BMD y obtenerlas de manera automatizada empleando la plataforma OraGenDel.

- **OBJETIVO 3 (OBJ. 3)**
  Evaluar la importancia de la base de datos LOVD no incluida en la plataforma OraGenDel.

- **OBJETIVO 4 (OBJ. 4)**
  Analizar y extraer el significado biológico de las variaciones relevantes que diferencian la DMD de la BMD, atendiendo al tipo de cambio y a la localización.

En el siguiente apartado de materiales se describirán las herramientas empleadas durante la ejecución del proyecto. Posteriormente, se desarrollará el método que se ha planteado con el primer objetivo (**Apartado 4**). De igual manera, a lo largo de las etapas de ese apartado, se establecerán los procedimientos necesarios para alcanzar el resto de los objetivos propuestos. Por último, se discutirán los resultados extraídos (**Apartado 5**), y finalmente, se plantearán las conclusiones y el trabajo futuro en esta línea de investigación (**Apartado 6**).

3. MATERIALES

Para la realización de este trabajo se emplearon materiales no tangibles, es decir, sistemas informáticos y datos extraídos de repositorios de información genómica. Debido a su carácter bioinformático, no se hizo uso de ningún laboratorio ni material experimental. En cuanto a la organización del apartado, en el **Subapartado 3.1**, se explican las herramientas *software* y lenguajes de programación empleados para el desarrollo del trabajo. Seguidamente, en el **Subapartado 3.2**, se profundiza en las bases de datos genómicas empleadas para la extracción de los datos, así como en sus características principales.

3.1. HERRAMIENTAS *SOFTWARE* Y LENGUAJES DE PROGRAMACIÓN

3.1.1. Oráculo Genómico de Delfos

La plataforma Oráculo Genómico de Delfos, también denominada OraGenDel, consiste en una herramienta *software* basada en un sistema de extracción y gestión de datos genómicos, escrita en el lenguaje de programación R y desarrollada en el Centro PROS-VRAIN, cuyo propósito es dotar de una arquitectura tecnológica al método SILE (explicado en el **Subapartado 1.1.1**), con el fin de permitir su automatización.

OraGenDel está formada por cuatro módulos: Hermes, Ulises, Delfos y Sibila. Cada módulo confiere el soporte necesario para implementar las diferentes etapas del método (Búsqueda, Identificación, Carga y Explotación). Además, cada uno de estos módulos también constituye una unidad funcional independiente, que se relaciona con los otros módulos mediante sus respectivos datos de entrada (*input*) y de salida (*output*).

Los 4 módulos que conforman la plataforma son los siguientes:

- **Hermes.** Este módulo se corresponde con la primera etapa del Método SILE: “Búsqueda”. Se encarga de conectar las fuentes de datos externas que proveen los datos crudos a la plataforma (Ensembl, ClinVar y GWAS Catalog). Tras incorporar estos datos de variaciones, se integran en un fichero de formato único que será el que utilice el siguiente módulo como *input.*
- **Ulises.** El paso del Método SILE con el que se corresponde es el segundo: “Identificación”. Su función principal es mensurar la calidad de los datos “ómicos” obtenidos mediante el empleo de un algoritmo que determina qué variaciones son relevantes para el diagnóstico de una enfermedad. Este algoritmo se encuentra actualmente en continua evolución y se explica en profundidad en el **Subapartado 4.3** del Método. En cuanto al *output* generado, se trata de un *set* o conjunto de variaciones cuya información correspondiente se encuentra asociada a las mismas.
- **Delfos.** Se corresponde con la tercera etapa del Método SILE: “Carga”. Este módulo se ocupa de proporcionar una interfaz para cargar el *output* de Ulises en la BBDD propia de la plataforma.
- **Sibila.** Está correlacionado con la última fase del Método SILE: “Explotación”. Su labor consiste en mostrar los datos almacenados en Delfos al usuario y proveer al mismo de las herramientas necesarias para la extracción de conocimiento (p.ej., proporcionar una representación gráfica con los distintos tipos de mutaciones que presenta cada variación).

Tanto Hermes como Ulises se ejecutaron mediante las funciones desarrolladas por el equipo en lenguaje R. La versión de R que se empleó fue la 4.0.3 y su visualización se llevó a cabo en el entorno proporcionado por RStudio v.1.2.5042. En cuanto a los módulos Delfos[1] y Sibila[2], estos disponían de una interfaz web para usuarios a la cual se podía acceder mediante un enlace o URL desde cualquier navegador.

A lo largo del **Apartado 4** se muestra en más detalle el funcionamiento de cada módulo, así como también las funciones requeridas en cada caso.

3.1.2. Python

Python es un lenguaje de programación de código legible y versátil, frecuentemente aplicado al ámbito del análisis bioinformático.

Para el desarrollo de una parte del código empleado, se recurrió al uso de este lenguaje de programación en su versión 3.9. Como entorno de programación se utilizó Visual Studio Code v.1.57.1, ejecutado por Anaconda Navigator 1.10.0.

Los *scripts* o programas creados con Python se encuentran explicados a modo de manual de usuario en el **Anexo 1**. Asimismo, en el **Apartado 4**, se muestra qué objetivo se alcanza con cada uno de ellos, así como el planteamiento utilizado para su desarrollo.

3.2. BASES DE DATOS GENÓMICAS

Una base de datos es una colección de información organizada que se caracteriza por permitir un fácil acceso, gestión y actualización de la misma. Las BBDD genómicas, por tanto, adoptan la definición anterior pero enfocada hacia la información genómica: colecciones de secuencias, mapas, ensamblajes, anotaciones, variaciones, etc. (Hutchings, 2020).

En relación con la problemática comentada anteriormente en el **Apartado 1** sobre la gestión de la inabarcable cantidad de datos genómicos, la editorial NAR (*Nucleic Acids Research*) de Oxford University Press publica todos los años un balance en el que analiza el número total de bases de datos existentes en el momento relacionadas con la biología y la bioinformática. Con respecto a los 1552 repositorios que había en 2014 (Fernández-Suárez *et al*., 2014), a día de hoy, el número asciende a 1641 (Rigden y Fernández, 2021), y la cifra va en aumento año tras año.

Esto último, muestra la notoria dificultad que viene implícita en la búsqueda de fuentes de información de las que extraer conocimiento y justifica el hecho de que para la búsqueda de variaciones genéticas relevantes en este trabajo se haya recurrido a más de un repositorio a partir del cual obtener datos. Concretamente, se han empleado 4 BBDD de variaciones genéticas: ClinVar, Ensembl, GWAS Catalog y LOVD, las cuales se explicarán a continuación.

[1] https://genomics-hub.pros.dsic.upv.es:3048/
[2] https://genomics-hub.pros.dsic.upv.es:3049/

3.2.1. ClinVar

ClinVar[3] es una base de datos pública y gratuita que almacena variaciones genéticas y sus interpretaciones de significancia clínica. Está mantenida por la NCBI desde su creación en 2013. La información incluida en esta BBDD proviene de laboratorios de análisis clínicos, grupos de investigación, bases de datos específicas de una localización concreta del genoma, paneles de expertos y otros grupos. Cada *submitter,* o persona encargada de cargar la variación en el repositorio, adjunta también una serie de información asociada a esa variación (p.ej., método de análisis, interpretación de su significancia clínica, literatura asociada, etc.) (Landrum *et al*. 2018).

3.2.2. Ensembl

Ensembl[4] es una base de datos de acceso gratuito que integra datos experimentales y de referencia sobre anotación estructural de genes, variaciones y elementos reguladores, permitiendo también realizar estudios de genómica comparativa. Está centrada en el genoma de los vertebrados, aunque también incluye a otros organismos de relevancia científica, como *Mus musculus* o *Danio rerio* (Howe *et al*. 2021). Fue puesta en marcha en el año 2000 y actualmente forma parte del EMBL-EBI (*European Molecular Biology Laboratory - European Bioinformatics Institute*). La información de las variaciones que está incluida en esta BBDD proviene de otros repositorios como dbSNP, EVA (*European Variation Archive*) o DGVa (*Database of Genomic Variants archive*), en los cuales la información puede haber sido cargada por cualquier usuario y, por lo tanto, no cumplir con los estándares establecidos por Ensembl. Por esta razón, antes de integrarse en Ensembl, pasa por un proceso de curado (al contrario que en ClinVar), así como también por programas que predicen automáticamente la significancia clínica como SIFT o PolyPhen2 (Hunt *et al.* 2018).

3.2.3. GWAS Catalog

GWAS Catalog[5] es un repositorio genómico que reúne los resultados de todos los estudios GWAS publicados. Desde 2010, ha sido mantenido por EMBL-EBI en colaboración con el NHGRI (*National Human Genome Research Institute*) (Buniello *et al*. 2019). En el presente proyecto, no es especialmente relevante debido a las características de los fenotipos estudiados, destacando el hecho de que, al tratarse de enfermedades raras, el bajo número de pacientes no permite llevar a cabo estudios de asociación con la suficiente potencia estadística. El único estudio GWAS relacionado con la DMD es el que realizó el equipo de Weiss (2018).

3.2.4. LOVD

La base de datos de LOVD[6] o *Leiden Open Variation Database* alberga una gran colección de variaciones genéticas asociadas a ciertos genes en particular. Uno de los genes en los que se centra es *DMD*, por lo que para el estudio de la enfermedad de DMD y BMD, se trata de la base de referencia a nivel mundial para el personal clínico e investigador (Fokkema *et al*. 2011). Uno de los objetivos de este trabajo (OBJ.3) fue analizar la importancia de esta BBDD, para la cual no se podía emplear la plataforma OraGenDel debido a la falta de conectores con la misma, por lo que sus datos se analizaron mediante el empleo de *scripts* desarrollados en Python (**Anexo 1**).

[3] https://www.ncbi.nlm.nih.gov/clinvar/
[4] https://www.ensembl.org
[5] https://www.ebi.ac.uk/gwas/home
[6] https://www.lovd.nl/

4. MÉTODO DE IDENTIFICACIÓN DE VARIACIONES

El método utilizado en este trabajo para la identificación de variaciones genéticas que muestran una clara implicación en el desarrollo de las enfermedades de Duchenne y de Becker, es el resultado de la fusión de dos métodos ya existentes: el Método SILE (**Subapartado 1.1.1**), y el método de IAE de Spreeuwenberg (**Subapartado 1.1.2**). Esto último se muestra de manera gráfica en la **Figura 3**.

El Método SILE es realmente el motor del trabajo, pues su fin último es la obtención de variaciones genéticas relevantes, lo cual puede aplicarse a cualquier fenotipo. En cambio, el método de IAE abarca un área más extensa, permitiéndose de este modo su implementación en diversos ámbitos del conocimiento. Con todo, la razón que ha llevado a proceder con la integración de ambos métodos ha sido el propósito de dotar al primero del respaldo que confieren las explicaciones que permite generar el segundo. Dada la creciente influencia que están adquiriendo los sistemas autónomos de toma de decisión, en especial en el contexto de la MP, es necesario en este punto establecer un protocolo de actuación que asegure la transparencia y la confianza en los datos obtenidos.

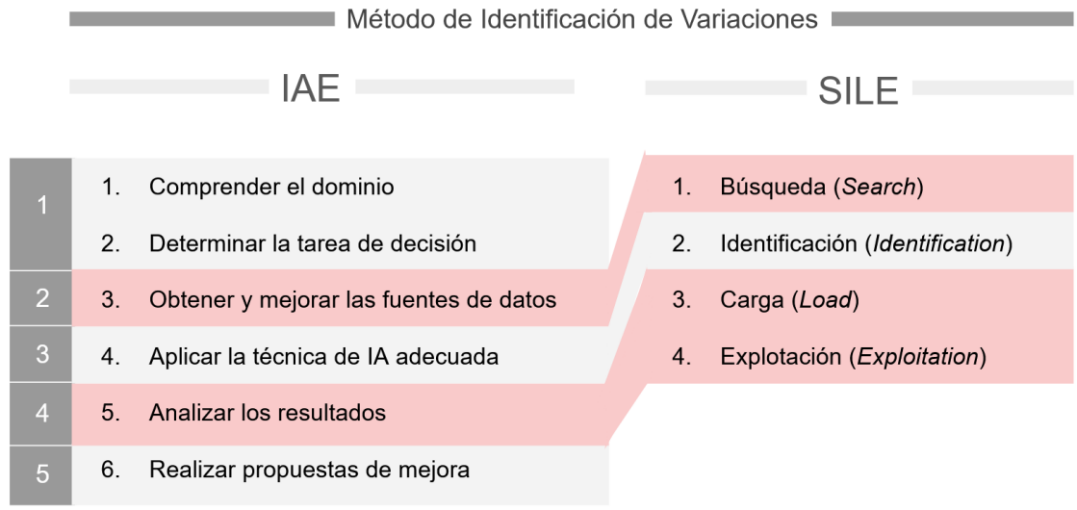

Figura 3. Correlación entre las diferentes etapas del método de IAE y SILE que da lugar al Método de Identificación de Variaciones empleado en el presente proyecto.

Las etapas que comprende este nuevo método son cinco en total, las cuales se desarrollan en los subapartados siguientes. La primera etapa del método de identificación de variaciones se corresponde con la primera y la segunda etapa del método de IAE, sin incluir al Método SILE (**Apartado 4.1**); la segunda etapa, con el tercer paso del método de IAE y el primero del Método SILE (**Apartado 4.2**); la tercera etapa, con el cuarto apartado del método de IAE y el segundo del Método SILE (**Apartado 4.3**). La cuarta etapa, en vez de relacionar un paso de cada método, unifica las dos últimas etapas del Método SILE junto con el quinto paso del método de IAE (**Apartado 4.4**). Y, por último, la quinta etapa se corresponde con la sexta del método de Spreeuwenberg (**Apartado 4.5**).

4.1. COMPRENSIÓN DEL DOMINIO Y DETERMINACIÓN DE LA TAREA DE DECISIÓN

La primera etapa del método empleado en el proyecto consistió en enmarcar el contexto del problema que se pretendía resolver y establecer de qué manera se iba a conseguir el objetivo planteado. Para ello, se requerían varios aspectos. El primero de ellos fue el conocimiento teórico de las bases de la enfermedad de DMD y de BMD, así como su relación con el Modelado Conceptual (MC) (**Subapartado 4.1.1**). El segundo aspecto consistió en el conocimiento de las secuencias de referencia sobre las cuales se anotaban las variaciones en las diferentes BBDD, para evitar cometer errores en la clasificación (**Subapartado 4.1.2**). Y, en cuanto al tercer aspecto, este vino en relación con la tarea de decisión, para la cual era crucial establecer un objetivo claro (**Subapartado 4.1.3**).

4.1.1. DMD y BMD en el contexto del Modelado Conceptual

Como se ha comentado en el **Apartado 1.2**, los fenotipos cuyas variaciones se han analizado en el presente estudio (DMD y BMD) se corresponden con enfermedades monogénicas que presentan un patrón de herencia recesivo ligado al cromosoma X, es decir, que albergan un claro origen genético. Esta afirmación, pese a su generalidad, ya permite establecer una correlación entre estos fenotipos y el Modelo Conceptual del Genoma Humano (MCGH), la cual se explica a continuación.

No obstante, antes de explicar dicha relación, es primordial conocer el concepto de MCGH, cuya idea aparece por primera vez en 2008 en la 27ª Conferencia Internacional de Modelado Conceptual (Pastor, 2008) y que se materializa con la tesis doctoral de J. F. Reyes (2018). El MCGH consiste en una representación visual y sistémica de los componentes que constituyen el genoma humano, interrelacionados entre sí según el proceso biológico concreto que da lugar a cada uno de ellos. En otras palabras, se trata de una estrategia basada en el MC, cuyo objetivo es organizar la información biológica desde la perspectiva de la ingeniería de Sistemas de Información (SI), con tal de abordar el problema de la dificultad de gestión de datos genómicos. En el **Anexo 2** se muestra la última versión del MCGH (v.3).

La ventaja que confiere el MCGH en el ámbito de la gestión de datos es que cubre la estructura completa del genoma, por lo que los datos almacenados en diferentes repositorios genómicos pueden conectarse fácilmente siguiendo las directrices del esquema (Palacio y López, 2018). Además, la adaptabilidad del MC confiere la posibilidad de abordar nuevos escenarios y evolucionar a medida que incrementa el número de fenotipos estudiados y que surgen nuevos retos.

Por consiguiente, una vez comprendido el concepto, ya se pueden enmarcar los dos fenotipos de estudio en el MCGH, el cual consta de cinco partes o vistas (“estructural” en verde, “transcripcional” en azul, “bibliográfica” en amarillo, “de variaciones” en naranja y “de rutas metabólicas” en rosa), como muestra la Figura 1B del **Anexo 2**. Sin embargo, para facilitar la visualización, en la **Figura 4** se representa, de manera esquemática, el dominio de trabajo en el que se va a basar el presente proyecto de investigación.

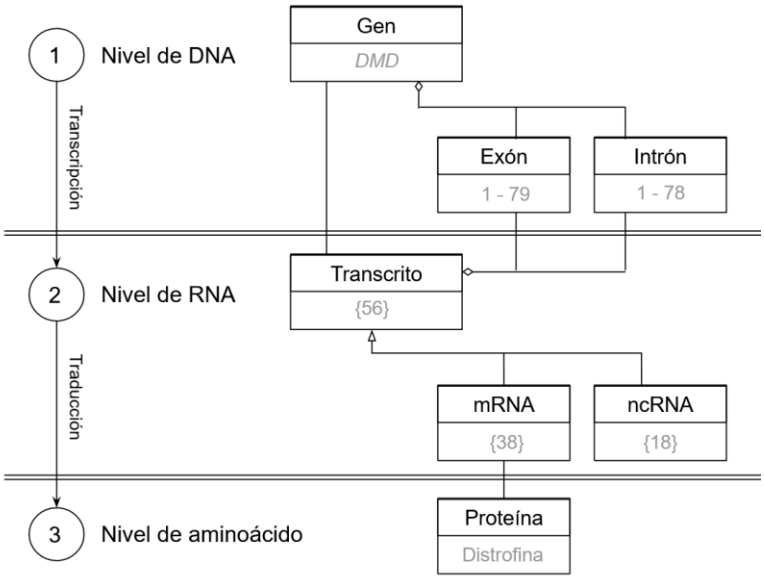

Figura 4. DMD y BMD dentro del MCGH. Fuente: imagen adaptada de García *et al.* 2020.

El diagrama de clases que se representa en la **Figura 4** se ha desarrollado siguiendo las normas básicas del MC, en concreto, del Lenguaje Unificado de Modelado (UML) simplificado (Ziemann *et al.* 2005).

En este caso, se muestran los tres niveles del proceso de expresión proteica, desde la secuencia genómica hasta la secuencia polipeptídica, pasando por el nivel transcripcional. En el primer nivel, se incluyen tres clases: "Gen", "Exón" e "Intrón". Las clases "Exón" e "Intrón" se conectan con la clase "Gen" por agregación (rombo blanco), lo cual indica que el gen tiene exones e intrones. En el segundo nivel, se incluyen otras tres clases: "Transcrito", "mRNA" (*messenger* RNA) y "ncRNA" (*non-coding* RNA), para las cuales ocurre lo mismo que en el nivel anterior, pero con la diferencia de que las clases "mRNA" y "ncRNA" se conectan a "Transcrito" por generalización (flecha blanca), lo cual indica que un transcrito puede ser codificante o no codificante, pero no incluir los dos. Además, para estas tres clases en concreto, se asocia un número entre corchetes, que indica el número de transcritos, mRNAs y ncRNAs del gen *DMD*, pero no sus nombres. Por último, en el tercer nivel, la clase "mRNA" se relaciona con "Proteína" mediante una unión simple.

En resumen, se ha expuesto el concepto de MCGH y su relación con los fenotipos de DMD y BMD, por lo que ambas enfermedades se han enmarcado en el dominio de trabajo.

4.1.2. Secuencias de referencia

Un hecho relevante que hay que tener en cuenta cuando se analizan variaciones genéticas, que contribuye a contextualizar el dominio de trabajo, es la secuencia de referencia sobre la que se mapea el material genético de la muestra extraída del paciente. Para ello, pueden emplearse como referencia: la secuencia genómica completa, o bien, la secuencia genómica codificante.

Cuando se trata de la secuencia genómica completa (a la que se hace referencia con una "g.") se está haciendo alusión al ensamblaje del genoma humano, ya sea la versión de 2009 ("GRCh37" o "hg19"), o la más reciente de 2014 ("GRCh38.p13"). También puede referirse a la secuencia genómica del cromosoma X (donde se encuentra el gen *DMD*), la cual también presenta dos versiones, "NC_000023.10" y "NC_000023.11", que se corresponden, respectivamente, con las versiones del genoma humano comentadas antes.

Hay ocasiones en que los alineamientos se efectúan sobre la secuencia genómica codificante (la cual se indica con una "c.") extraída de un transcrito concreto. En la actualidad, para el gen *DMD*, se han detectado más de 30 transcritos codificantes distintos en humanos, cada uno de los cuales codifica para varias isoformas, como se ha comentado en la Introducción (**Apartado 1.2**). Para los casos de Duchenne y de Becker, la isoforma más relevante es la que se expresa en el tejido muscular esquelético, para la cual su transcrito codificante es "NM_004006", según la NCBI, o "Dp427m". No obstante, la isoforma que se expresa en neuronas y cuyo transcrito codificante es "NM_000109" o "Dp427c", también es una de las más comunes y cuya alteración predispone también al desarrollo de estas distrofinopatías. Ahora bien, al igual que ocurría con el genoma de referencia, también existen distintas versiones de los transcritos de referencia (p.ej., "NM_004006.2", "NM_004006.3", etc.), lo cual dificulta la tarea de estandarización de las variaciones a analizar.

A fin de llevar a cabo un estudio riguroso de comparación de las variaciones en los distintos repositorios genómicos, fue importante identificar sobre qué secuencia de referencia (y su versión) se había establecido la anotación.

En el caso de las variaciones provenientes de la base de datos LOVD, se indicaba expresamente en la parte superior de la página web que las variaciones mostradas se habían descrito empleando el transcrito de referencia "NM_004006.2", es decir, la versión 2 del transcrito "NM_004006". Y, de manera adicional, se proporcionaba la localización ocupada por la variación en la secuencia genómica, tanto en la versión GRCh37 como en la GRCh38.

En cuanto a la BBDD de ClinVar, al no tener establecido un proceso de curado y selección sobre las variaciones cargadas por los usuarios, se observaron variaciones anotadas en base al transcrito "NM_004006" tanto en su versión 2 como en la 3. Luego, al igual que en LOVD, se proporcionaba la localización genómica que ocupaba la variación tanto en la versión GRCh37 como en la GRCh38.

La BBDD de Ensembl, a diferencia de las dos anteriores, permitía seleccionar el genoma de referencia empleado, por lo que todas las variaciones estaban anotadas en base a la versión más actual del genoma humano.

Por último, en GWAS Catalog, al provenir todas las variaciones que se analizaron en el proyecto de un único estudio (Weiss *et al.*, 2018), únicamente se empleó el genoma de referencia GRCh38.

4.1.3. Planteamiento del OBJETIVO 2

Una vez comprendido el contexto en que se desarrollaría el trabajo, fue determinante establecer un objetivo claro que confiriese una direccionalidad al proyecto. Esto es precisamente lo que proponía el OBJ. 2, en el cual se pretendía llevar a cabo la aplicación de un algoritmo de IA de clasificación para la identificación de variaciones genéticas relevantes asociadas a la DMD y a la BMD, haciendo uso de la plataforma

OraGenDel. Realmente, este era el objetivo primordial del trabajo: la obtención de estas variaciones relevantes asociadas a los fenotipos de estudio. No obstante, el OBJ. 2 apuntaba a la forma de materializar esta meta, que era mediante la aplicación del algoritmo que se explica en el **Apartado 4.3**.

4.2. OBTENCIÓN Y MEJORA DE LOS DATOS

Esta segunda etapa del método consistió en seleccionar las fuentes de información y en extraer los datos que estas albergaban, en este caso variaciones, con tal de proceder a la mejora de los mismos (**Subapartado 4.2.1**). Esto último, se realizó filtrando las redundancias que pudiesen haber y unificando el formato, para poder, de este modo, realizar comparaciones independientemente de la fuente de la que hubieran sido extraídos (**Subapartado 4.2.2**).

4.2.1. Extracción de las variaciones

Para la extracción de las variaciones de Ensembl, ClinVar y GWAS Catalog, se empleó el módulo Hermes de la plataforma OraGenDel. La manera de proceder consistió en ejecutar una serie de funciones propias de este módulo, en la consola de RStudio, que permitieron generar, en última instancia, un archivo por cada *set* las variaciones de cada una de las tres BBDD, en formato de valores separado por comas (CSV). Además, las funciones permitían indicar el fenotipo para el cual se pretendía extraer la información, por lo que este proceso se realizó tanto para el fenotipo de Duchenne como para el de Becker.

En cuanto a las variaciones de la base de datos de LOVD, la cual no se encontraba entre las opciones de la plataforma OraGenDel, se extrajeron directamente de la interfaz web del propio repositorio. Seguidamente, se guardaron en un fichero Excel (XLSX) y se exportaron al formato de valores separados por tabuladores (TSV), para poder ser gestionadas posteriormente mediante el empleo de *scripts* en Python.

Con las variaciones extraídas de este último repositorio no se pudo realizar una distinción entre fenotipos, dado que en la información asociada a cada variación no existían los datos necesarios para llevar a cabo tal clasificación. Además, todas las variaciones que se almacenaban en dicha BBDD se sabe que se encontraban en el gen *DMD*, pero no se especificaba información referente a la enfermedad que producían (p.ej., DMD, BMD, etc.). No obstante, dado que el transcrito de referencia empleado es el que se corresponde con la isoforma de la distrofina expresada en el músculo (“NM_004006.2”), las variaciones estaban asociadas principalmente a DMD y BMD.

4.2.2. Filtrado y uniformización de las variaciones

Una vez realizada la extracción con OraGenDel de las variaciones de Ensembl, ClinVar y GWAS Catalog por separado, se empleó otra función del módulo Hermes que permitía integrar en un único fichero todas las variaciones, así como también eliminar las redundancias que pudiesen haber de una BBDD a otra. La forma de poder comparar las variaciones de cada repositorio de manera sistemática se conseguía gracias a la uniformidad en el nombre de las variaciones que permitía generar dicha función. Este nombre atendía a la siguiente estructura:

'Localización : Cambio (Ensamblaje)'

El campo "Localización" está formado por el cromosoma en que se encuentra la variación y por la posición exacta en pares de bases o en número que ocupan los aminoácidos en la cadena polipeptídica: *'Chr {chr_name} : {g | c | p} . {start | ?} – {end | ?}'*. Asimismo, se hace una distinción entre DNA (*Deoxyribonucleic Acid*) genómico ("g."), cDNA ("c.") y proteínas ("p."), en función de la información de la que disponga la variación.

El siguiente campo, "Cambio", representa la secuencia original frente a la de la variación. Por lo general, se suele referir a esto último como "alelo de referencia" y "alelo alternativo". No obstante, en este campo, existe una restricción creada con tal de facilitar la visualización de la información. Esta consiste en que una vez se superan los 10 caracteres (referido a pbs), tanto si se trata únicamente del alelo de referencia como del alternativo, este campo toma el valor "*longchange*". El esquema es el siguiente: '*{{ref | ?} > {alt | ?} | longchange}'*.

Por último, en cuanto al campo "Ensamblaje", este suele tomar los valores del genoma de referencia que haya sido utilizado para la anotación ("GRCh37" o "GRCh38"). Sin embargo, también puede hacer referencia a una secuencia de cDNA obtenida a partir de un transcrito codificante (p.ej., "NM_004006.2") o a una secuencia proteica (p.ej., "NP_003997.1").

Prosiguiendo con el resto de fuentes de datos, en lo referente a la información extraída de LOVD, se procedió con el filtrado de las redundancias y con la uniformización de las variaciones bajo un nombre con formato OraGenDel. La realización de estos pasos fue posible gracias al desarrollo de los *scripts* que se explican en el primer anexo (**Anexo 1**).

4.2.3. Planteamiento del OBJETIVO 3

Como se comentaba en el **Apartado 2** de Objetivos y en el **Subapartado 3.2.4** de Materiales, uno de los objetivos del trabajo (OBJ.3) consistía en analizar la importancia del repositorio de datos genómicos de LOVD. Este objetivo surgió, precisamente, a raíz de la aplicación de esta segunda fase del método, concretamente, al plantearse de qué BBDD se pretendía extraer las variaciones. Tras optar por incluir esta BBDD al estudio, la primera observación fue que la plataforma automatizada OraGenDel no ostentaba los conectores necesarios para poder extraer la información de esta BBDD. Este hecho, llevó a la necesidad de realizar un análisis manual de las variaciones de este repositorio, que finalmente terminó por automatizarse mediante *scripts* desarrollados con el lenguaje de programación Python (**Anexo 1**).

En relación con las variaciones de LOVD resultantes del proceso de filtrado y uniformización mediante *scripts* comentado en el subapartado anterior, se llevó a cabo una comprobación de la validez de dichos resultados. Para ello, se evaluó la capacidad que presentaban los *scripts* realizados con Python para extraer las variaciones y unificarlas bajo el mismo formato. La forma de llevar a cabo dicha validación fue extraer de forma manual desde la interfaz web de los repositorios ClinVar y Ensembl, las variaciones asociadas a los dos fenotipos. Seguidamente, los ficheros "crudos" se modificaron mediante los *scripts* para eliminar las redundancias y unificar el formato del nombre de cada variación bajo la estructura propia de OraGenDel: *'Localización : Cambio (Ensamblaje)'*. Por último, se calculó el ratio de cobertura del fichero obtenido haciendo uso de los *scripts* frente a las variaciones obtenidas automáticamente mediante la plataforma OraGenDel. En caso de mostrar una cobertura del 100%, indicaría que los

scripts que se desarrollaron para extraer las variaciones de LOVD serían capaces de extraer la misma información que podría obtener la plataforma OraGenDel en el caso de disponer de los conectores necesarios con esta BBDD. Los resultados se muestran en el **Apartado 5.3**.

4.3. APLICACIÓN DE LA TÉCNICA DE IA ADECUADA

La tercera etapa del método se basó en la aplicación de un algoritmo de IA de clasificación de variaciones genéticas, el cual se explica en el **Subapartado 4.3.1**. Las variaciones que se clasificaron empleando este algoritmo fueron las que habían sido extraídas por el módulo Hermes de OraGenDel en el apartado anterior, es decir, las variaciones de Ensembl, ClinVar y GWAS Catalog. Es importante resaltar que esta etapa es la que engloba a la segunda fase del Método SILE (Identificación), la cual es prácticamente la más importante.

4.3.1. Algoritmo de clasificación (Ensembl, ClinVar y GWAS Catalog)

La clasificación de las variaciones se realizó de manera automática mediante la ejecución de una función implementada en el módulo Ulises dentro de la plataforma OraGenDel. Esta clasificación se llevó a término siguiendo el algoritmo de clasificación propuesto en el Método SILE (León Palacio, 2019).

En cuanto a los resultados de la clasificación, existen 5 niveles o categorías según el tipo de relevancia clínica que presentan: 1) Aceptada con Evidencia Fuerte (*ACCEPTED-STRONG*), 2) Aceptada con Evidencia Moderada (*ACCEPTED-MODERATE*), 3) Aceptada con Evidencia Limitada (*ACCEPTED-LIMITED*), 4) En Seguimiento (*TO FOLLOW UP*) y 5) Rechazada (*DISCARDED*). En la **Figura 5**, se muestra la representación del algoritmo en su totalidad.

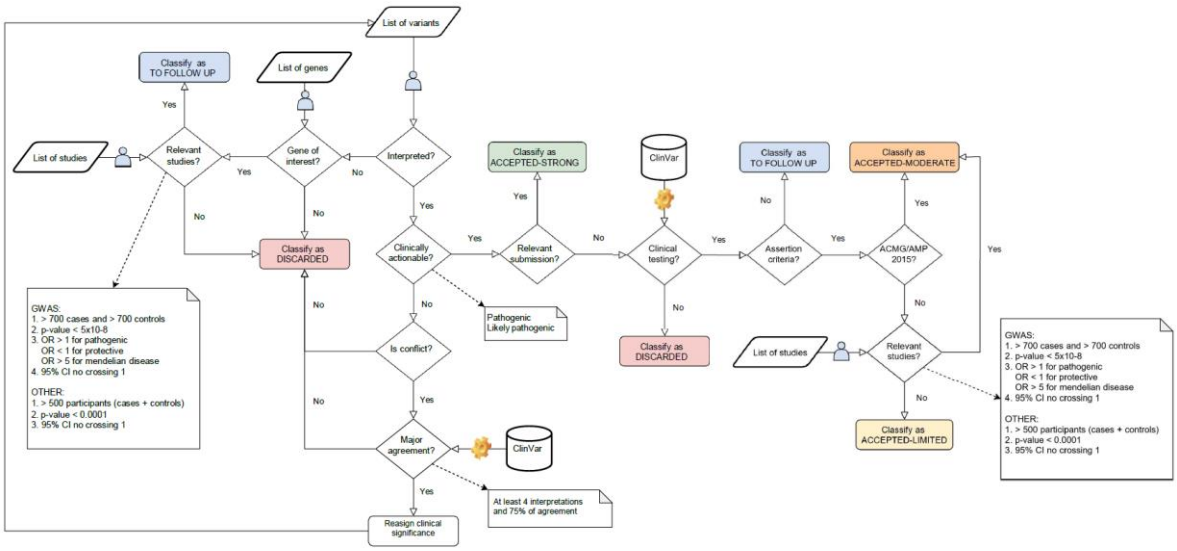

Figura 5. Algoritmo de clasificación de variaciones genéticas relevantes propuesto por el Método SILE. Fuente: León Palacio, 2019.

Siguiendo el diagrama que representa el algoritmo de clasificación, este comienza cuando se le proporcionan tres tipos de datos al sistema (en este caso a la plataforma OraGenDel): una lista de estudios, una lista de variaciones y una lista de

genes. La lista de estudios y la de variaciones se obtuvieron a partir del fichero integrado que había sido generado en la etapa anterior. La lista de genes, en cambio, fue proporcionada por el usuario.

En referencia a esto último, debido a que podía darse el caso de que hubiera variaciones que se encontrasen en más de un gen, como ocurre en las enfermedades multigénicas, existía una variable que permitía almacenar una lista de genes que se sabía, por literatura, que estaban implicados. En este caso, se buscaron variaciones principalmente en el gen *DMD*, pues los fenotipos que se analizaron en el presente proyecto estaban directamente relacionados con alteraciones en dicho gen.

Ahora bien, se sabe que los estudios GWAS pueden mostrar asociación entre fenotipos y variaciones que se encuentran en diversos genes, por lo que podría ocurrir que el usuario que pretendiese extraer variaciones relevantes, no estuviese al tanto de todos los genes relacionados. Este hecho, sin embargo, no se consideraba un impedimento para que las variaciones pudieran ser clasificadas de manera normal, incluso hasta para ser aceptadas con evidencia fuerte en caso de haber cumplido con los requisitos. En el caso de la DMD, únicamente existía un estudio tipo GWAS, en el cual se hallaron nuevos genes que en bibliografía previa no se habían relacionado con la enfermedad (p.ej., *RGS6*, *XYLT1*, *LINC02141*).

Con respecto a la asignación a cada variación a las categorías comentadas anteriormente, se va a dividir el diagrama de flujo de la **Figura 5** en tres partes para facilitar su comprensión. La primera vista se muestra a continuación (**Figura 6**).

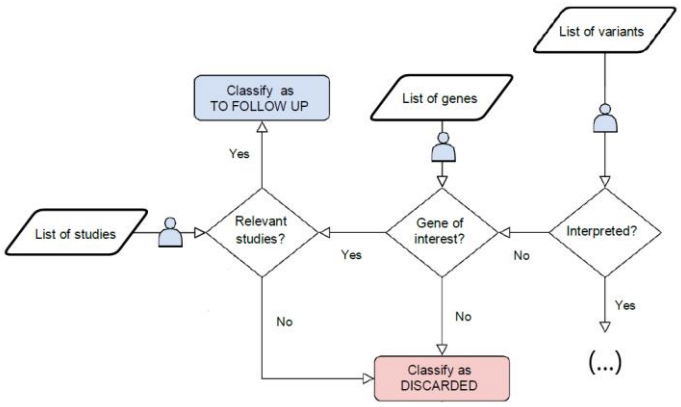

Figura 6. Vista 1 del algoritmo de clasificación.

Cuando la variación de la lista de variaciones no había sido interpretada, es decir, cuando no se indicaba su significancia clínica, y tampoco se encontraba en ninguno de los genes de interés, quedaba descartada (*DISCARDED*). Si, por el contrario, se encontraba en uno de los genes de interés de la lista y, además, presentaba estudios relevantes, se clasificaba como *TO FOLLOW UP*. Ahora bien, en el caso de no presentar estudios relevantes, aún si se encontraba en un gen de interés, quedaba descartada. En este punto, es importante mencionar que un estudio se considera relevante o de interés cuando cumple ciertas características, siendo la principal de ellas provenir de GWAS Catalog y, en algunos casos, de Ensembl. El resto de las características relevantes estaban relacionadas con el número de participantes y estadísticas asociadas, tal y como se observa en la **Figura 5**. En este punto es relevante remarcar que las variaciones del único estudio GWAS llevado a cabo con pacientes de DMD no pasó los requerimientos establecidos por el algoritmo, debido al bajo número de participantes, por lo que todas estas variaciones fueron clasificadas como *REJECTED*.

Continuando con el flujo del diagrama, que se representa ahora en la vista 2 (**Figura 7**), podía ocurrir que la variación hubiese sido interpretada y, pese a ello, ser clasificada como descartada.

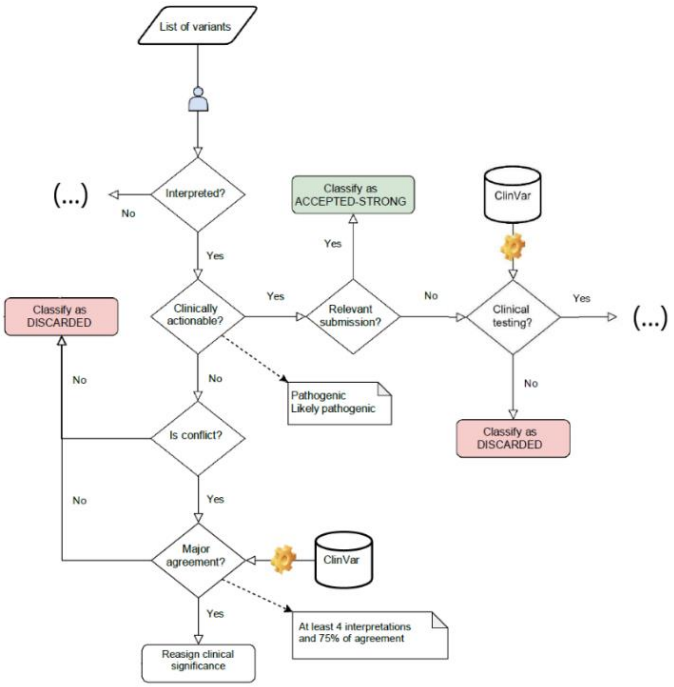

Figura 7. Vista 2 del algoritmo de clasificación.

Esto último ocurría cuando no era clínicamente accionable o, en otros términos, cuando no había sido clasificada como "Patogénica" ni como "Probablemente patogénica". Además, cabía la posibilidad de que sí fuera accionable pero no se hubiese indicado el método de evaluación empleado, en cuyo caso, quedaba descartada también. Sin embargo, en el caso de haber sido interpretada la variación y de ser, además, clínicamente accionable, podía pasar a ser clasificada de múltiples maneras, como muestra la tercera vista de la representación del algoritmo (**Figura 8**).

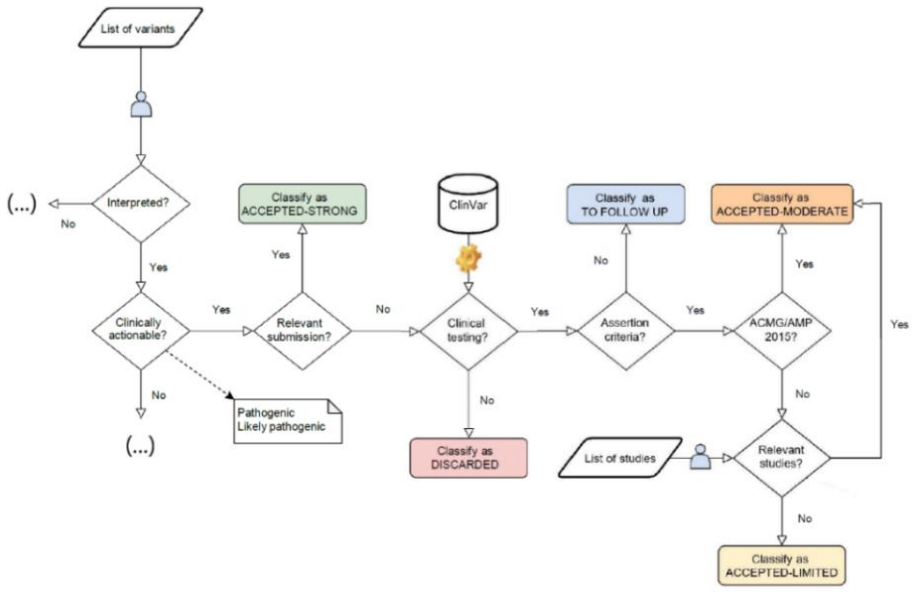

Figura 8. Vista 3 del algoritmo de clasificación.

Si la fuente era relevante o *relevant submission* (p.ej., panel de expertos, guía de práctica clínica) se clasificaba como *ACEPTED-STRONG*. En caso de no ser relevante y de no haber sido testada clínicamente, se clasificaba como *DISCARDED*. Siguiendo con el diagrama, si la variación había sido testada clínicamente pero no se indicaba el criterio de evaluación empleado por el *submitter* (p.ej., ACMG Guidelines 2015, Mendelics Assertion Criteria 2017, etc.), era clasificada como *TO FOLLOW UP*. Luego, en caso de sí indicarse el criterio de evaluación y de pertenecer este último, además, a la guía establecida en 2015 por la ACMG (*American College for Medical Genetics and Genomics*) o la AMP (*Association for Molecular Pathology*), es decir, si tenía asignado alguno de los términos siguientes: "patogénica", "probablemente patogénica", "significancia indeterminada", "probablemente benigna" o "benigna" (Richards *et al*. 2015) se clasificaba como *ACCEPTED-MODERATE*. Y, por último, todas aquellas variaciones que cumplían lo anterior mencionado excepto tener asociada una significancia clínica de acuerdo con los estándares de la ACMG/AMP y que, además, no presentasen estudios relevantes, se clasificaban como *ACCEPTED-LIMITED*.

4.3.2. Corrección del *output* generado

Tras obtener la clasificación, el fichero unificado con las variaciones se hizo pasar por una función de Hermes que generaba una serie de avisos indicando qué variaciones de las obtenidas presentaban inconsistencias, como se muestra en la tabla siguiente (**Tabla 1**).

Tabla 1. Tipo de alertas generadas por la plataforma y el número de veces que aparecen en los fenotipos de estudio.

	Nº	Descripción	Duchenne	Becker
"ERROR!"	1	*The name of the variant must be specified*	125	3
	2	*The name of the assembly must be specified*	41	4
	3	*There is more than one start coordinate*	46	1
	4	*There is more than one end coordinate*	100	3
	5	*The chromosome must have only one name*	3	0
	6	*Alternative allele must have only one value*	12	0
"Warning"	7	*Ref and alt alleles are missing*	258	12
	8	*Ref and alt alleles are the same*	3	3
	9	*The variant type is missing*	731	1
	10	*The alternative allele must be unique to generate the name of the variant with id=rs149640*	12	0
	11	*The chromosome must be unique to generate the name of the variant with id=rs12524310*	3	7
			1334	27

Como estos avisos aparecían en la consola de R, se empleó la función "*sink*" para redirigir el *output* hacia un archivo de texto. Además, para poder manejar el fichero con las variaciones, se convirtió al formato CSV. Seguidamente, se evaluaron manualmente las variaciones que tengan *warnings* o errores asociados.

La manera de resolver las alertas de tipo "ERROR!" fue mediante el empleo de los atributos que venían asociados a cada variación y que se encontraban en el mismo archivo. Por ejemplo, para resolver el Error tipo 1, el cual indicaba que no había ningún valor en la columna de nombre de la variante ("*variant_name"*) del fichero, se acudió a la información que contenían las columnas adyacentes: "*chromosome"*, "*alt"*, "*ref"*, *"start"* y "*end"*. En la **Tabla 2** se muestra un ejemplo.

Tabla 2. Ejemplo de error nº1 y su corrección.

	variant_name	*assemblies*	*chr.*	*alt*	*ref*	*start*	*end*
Error (nº 1)	-	GRCh38|GRCh37	X	G	T	31875190	31875190
Corrección	ChrX:g.31875190-31875190:T>G(GRCh38)	GRCh38|GRCh37	X	G	T	31875190	31875190

Como se observa en la tabla de arriba (**Tabla 2**), la primera fila representa una variación a la cual le faltaba el nombre. En cambio, en la segunda fila, se muestra el resultado tras la corrección.

Con respecto a las alertas tipo "*Warning*", la forma de resolverlas se basó en la búsqueda de información en la propia BBDD de la que provenía la variación. En este caso, el problema a resolver no era tanto de tipo formal sino más bien de información contradictoria o falta de la misma. Un ejemplo que muestra muy bien este hecho es la alerta tipo 8, para la cual tanto el alelo de referencia como el alternativo eran el mismo. La solución a este problema se consiguió buscando la variación en el repositorio correspondiente y anotando adecuadamente los alelos.

Continuando con este tipo de alertas, si bien su resolución consistió principalmente en acudir a fuentes externas que no fueran el propio fichero CSV, en ocasiones, había atributos de la variación en el propio archivo cuya información permitía su resolución. Por ejemplo, el *warning* tipo 9, que aparecía en repetidas ocasiones para ambos fenotipos y el cual indicaba que no se encontraba el tipo de variación, se resolvió acudiendo a las columnas de los alelos ("alt" y "ref"), como se observa en la **Tabla 3**.

Tabla 3. Ejemplo de distintos tipos de variaciones en el fenotipo DMD.

variant_type	*variant_name*	*assemblies*	*chr*	*alt*	*ref*	*start*	*end*
SNP	ChrX:g.32491520-32491520:T>G(GRCh38)	GRCh38	X	G	T	32491520	32491520
Deleción	ChrX:g.32849736-32849736:G>?(GRCh38)	GRCh38	X	-	G	32849736	32849736
Inserción	ChrX:g.32454659-32454660:?>TA(GRCh38)	GRCh38	X	TA	-	32454659	32454660
Duplicación	ChrX:g.32573781-32573782:TCTT>TCTTTCTT(GRCh38)	GRCh38	X	TCTTTCTT	TCTT	32573781	32573782
Microsatélite	ChrX:g.32216962-32216963:TGTGT>TGT (GRCh38)	GRCh38	X	TGT	TGTGT	32216962	32216963

En dicha tabla, se muestran distintos tipos de variaciones, que se indican en la primera columna, junto con los atributos correspondientes a las mismas, que se observan en el resto de columnas. El *warning* tipo 9 comentado anteriormente consistía en no tener ninguna información anotada en la primera columna, lo cual se podía resolver extrayendo información del resto de columnas.

Otro error que es interesante comentar es el nº10, que surgía como indicador de que existía más de un alelo alternativo. Este error ponía de manifiesto la diferencia en cuanto al formato de nombre que presentaba la plataforma OraGenDel frente a otro tipo de formatos (como p.ej., la nomenclatura establecida por la HGVS o *Human Genome Variation Society*). Mientras que esta última nombra a cada variación en función de la localización que ocupa en el genoma, independientemente de si existen uno o más alelos asociados a ese locus, OraGenDel determina que cada variación únicamente puede tener un alelo alternativo. En caso de tener más de un alelo alternativo, esta se considera otra variación diferente, como se observa en la **Tabla 4**.

Tabla 4. Ejemplo de error nº10 y su corrección.

	variant_name	*chr*	*alt*	*ref*	*start*	*end*
Error (nº 10)	-	X	A|C|T	G	77654786	77654786
Corrección	ChrX:g.77654786-77654786:G>A(GRCh38)	X	A	G	77654786	77654786
	ChrX:g.77654786-77654786:G>C(GRCh38)	X	C	G	77654786	77654786
	ChrX:g.77654786-77654786:G>T(GRCh38)	X	T	G	77654786	77654786

Estas alertas permitieron corregir los errores y preparar el archivo para su correcta carga en el repositorio de Delfos, el tercer módulo de OraGenDel, el cual se explica en el siguiente apartado.

4.4. ANÁLISIS Y EXPLICACIÓN DE LOS RESULTADOS

La cuarta etapa de esta metodología consistió en realizar una serie de análisis a partir de los resultados obtenidos. No obstante, dado que el apartado de Resultados y Discusión es el que se suele destinar a tal fin, en este subapartado se incluye el planteamiento de los análisis que se realizaron. En el **Apartado 5**, en cambio, se discuten los resultados obtenidos.

4.4.1. Carga y Explotación

El propósito de la fase de *Load* (Carga) consistía en cargar la información obtenida tras completar las fases de Búsqueda e Identificación del Método SILE haciendo uso del módulo de Delfos. En primer lugar, se accedió a la plataforma mediante la interfaz web. A continuación, se completaron los campos requeridos: "*username/email*" y "*password*". Y, por último, se procedió a cargar las variaciones, subiendo al repositorio el archivo que había sido corregido en el paso anterior, en formato CSV.

Para la fase de *Explotation* (Explotación), se accedió a Sibila mediante URL con el navegador y se realizó, por separado, la búsqueda de los fenotipos DMD y BMD. Esto condujo a una página web donde se mostraba la lista con las variaciones cargadas, así como también, gráficas circulares donde se clasificaban las variaciones según el tipo de cambio. Estos resultados se muestran en el cuarto subapartado del **Apartado 5**.

4.4.2. Descripción del OBJETIVO 3

Retomando lo explicado en el último párrafo del **Subapartado 4.2.2**, en el cual se exponía de qué manera se realizó la validación de los *scripts* de Python con el fin de garantizar el buen manejo de los datos extraídos de LOVD, a continuación, se muestra el resultado obtenido a partir de dicha validación.

Al comparar cuántas de las variaciones, que habían sido previamente obtenidas tras el filtrado y la uniformización de los nombres mediante *scripts*, habían sido también extraídas y filtradas por la plataforma OraGenDel, se calculó una cobertura del 96.72% para BMD y del 99.75% para DMD. Las variaciones en común para la enfermedad de Becker fueron 59/61 y, para la enfermedad de Duchenne, 810/812. Cabe destacar, no obstante, que los cálculos de las coberturas se realizaron sobre el total de variaciones extraídas por la plataforma, ya que con los *scripts* se obtenía un número mayor de variaciones para cada fenotipo (1722 variaciones para DMD y 92 para BMD) y el cálculo de la cobertura nunca hubiera podido alcanzar el 100%.

La razón por la que había más variaciones extraídas con los *scripts* se debía a que la manera de clasificar las variaciones únicamente estaba basada en descartar aquellas cuya significancia clínica fuese distinta de "patogénica" o "probablemente patogénica". Por ello, todas aquellas variaciones que podrían haber sido rechazadas en base a otros criterios (como p.ej., aquellas que no habían sido testadas clínicamente), no se descartaron. A pesar de esto, los resultados de la validación indicaron que, si bien no se habían descartado todas las variaciones que podrían haber sido descartadas, las variaciones importantes (las que sí habían sido aceptadas por OraGenDel) sí que estaban, lo cual generaba seguridad y contribuía a aumentar las garantías de dichos *scripts* para posteriores usos.

Tras la confianza profesada hacia los *scripts* atendiendo a los resultados de la validación, se pudo continuar con la consecución del OBJ. 3, en el cual se pretendía analizar la importancia de la BBDD de LOVD para la obtención de variaciones relevantes asociadas a ambos fenotipos. Para ello, se realizó la siguiente comprobación: se calculó la cobertura de las variaciones de la BBDD de LOVD frente a las de la plataforma OraGenDel. Las variaciones de esta última provenían de Ensembl y de ClinVar, pero no de GWAS Catalog, dado que fueron rechazadas por el algoritmo en el paso anterior. Los resultados obtenidos tras el experimento y respaldados por los expertos del Grupo de Investigación en Biomedicina Molecular, Celular y Genómica del IIS La Fe de Valencia, entre ellos el Dr. José M. Millán, se muestran en el **Apartado 5.3**.

4.4.3. Planteamiento del OBJETIVO 4

El OBJ. 4 estaba centrado en la extracción del significado biológico propio de las variaciones que se correspondían únicamente con la DMD y de las que se correspondían sólo con la BMD. Para extraer este significado biológico, se analizaron dos variables: el tipo de cambio y la posición de las variaciones. La realización de ambos análisis se llevó a cabo mediante *scripts* desarrollados con Python (**Anexo 1**).

Con el tipo de cambio, se pretendía analizar cuál era la causa mutacional más frecuente que conducía al desarrollo de cada enfermedad. Los tipos de mutaciones abarcaban desde variaciones de un único nucleótido (o SNV, por sus siglas en inglés: "*Single Nucleotide Variant*") hasta grandes deleciones o duplicaciones, pasando por inversiones e inserciones, entre otras. Un hecho relevante a destacar con respecto a las SNV es que estas se pueden desglosar en varios subtipos distintos atendiendo a la alteración traduccional que derive de las mismas: mutaciones "*nonsense*" o sin-sentido (producen un codón de *STOP* prematuro que detiene la síntesis proteica antes de tiempo) y mutaciones "*missense*" o de cambio de sentido (la variación da lugar a un codón distinto del que había anteriormente, el cual sí codifica para un aminoácido que es diferente del original).

En cuanto a la localización, se estableció una relación entre el lugar del genoma en que se producía la alteración y la estructura proteica con la que se correspondía. Para ello, se analizó el número de variaciones que se encontraban repartidas a lo largo del gen *DMD*. Un hecho relevante es que las variaciones se ordenaron siguiendo la estructura de los 79 exones, pero también en base a los intrones adyacentes. Esto es debido a que las variaciones intrónicas pueden, en ocasiones, alterar secuencias aceptoras o dadoras de *splicing*, dando lugar a alteraciones que se ven reflejadas en la fase traduccional. Por último, se realizó la correspondencia entre las secuencias genómicas donde se acumulaban más variaciones con sus respectivas regiones en la proteína distrofina, haciendo uso de la herramienta BLASTX y realizando búsquedas en la página web de UniProtKB.

Los resultados generados a partir de este análisis se exponen en el **Apartado 5.4** de Resultados y Discusión.

4.5. REALIZACIÓN DE PROPUESTAS DE MEJORA

La última etapa del método consistió en la elaboración de una serie de propuestas enfocadas al refinamiento y mejora del proceso de identificación de variaciones relevantes. Ahora bien, con tal de fundamentar las propuestas en base a los resultados obtenidos, este tema se desarrolla en mayor profundidad en el **Apartado 5.5**.

5. RESULTADOS Y DISCUSIÓN

Este apartado se estructura siguiendo el orden que presenta el **Apartado 2** de Objetivos. A lo largo de cada uno de los cuatro siguientes subapartados (**5.1**, **5.2**, **5.3** y **5.4**) se exponen los resultados obtenidos tras la aplicación del método, a la vez que se procede a la discusión de los mismos. El último subapartado (**5.5**), en cambio, está dirigido a la exploración de las diversas propuestas de mejora que han ido surgiendo durante la realización del presente proyecto TFG y que constituyen en sí mismas una parte del método que se ha empleado en el trabajo de investigación.

5.1. RESULTADOS Y DISCUSIÓN DEL OBJETIVO 1

El OBJ. 1 consistía en enmarcar el Método SILE dentro de un método de IAE, lo cual implicaba la aparición de un método no existente hasta la fecha. Este objetivo se ha cumplido con el desarrollo del propio apartado de Método (**Apartado 4**). En esta línea, los cinco subapartados que vertebran dicho apartado se corresponden con las etapas del método de identificación de variaciones relevantes.

El resto de los objetivos planteados en el trabajo (OBJ. 2, OBJ. 3 y OBJ. 4) se consiguieron resolver también, como se explica en los subapartados siguientes, haciendo uso del método creado a partir del primer objetivo.

5.2. RESULTADOS Y DISCUSIÓN DEL OBJETIVO 2

El OBJ. 2 se centraba en aplicar el algoritmo de identificación de variaciones relevantes asociadas a la DMD y a la BMD, para obtenerlas de manera automatizada mediante la plataforma OraGenDel. Dicho algoritmo se ha explicado en detalle en el apartado anterior (**Subapartado 4.3.1**). En cuanto a los resultados obtenidos tras su aplicación, en la **Tabla 5** se muestra el número de variaciones que fueron clasificadas en cada categoría.

Tabla 5. Resultados de la clasificación obtenidos tras la aplicación del algoritmo de IA.

	Duchenne		Becker	
	Número	%	Número	%
ACCEPTED-STRONG	0	0	0	0
ACCEPTED-MODERATE	43	1.67	7	5.74
ACCEPTED-LIMITED	757	29.33	40	32.79
TO FOLLOW UP	15	0.58	14	11.48
DISCARDED	1766	68.42	61	50
	2581	100	122	100

Como se observa en la **Tabla 5**, la primera diferencia se encuentra en la cantidad de variaciones extraídas para la enfermedad DMD y para la BMD. Para la DMD, se extrajeron 2581 variaciones de los repositorios de ClinVar, Ensembl y GWAS Catalog; mientras que para la BMD, 122 variaciones. Además, de este último, no se encontraron estudios GWAS, por lo que las variaciones procedían de ClinVar y Ensembl únicamente. Asimismo, el hecho de encontrar más variaciones asociadas a la DMD que a la BMD, ambas tratándose de enfermedades con un fenotipo similar, pone de manifiesto la falta

de estudios con respecto de la enfermedad de Becker, un problema que se pretendió abordar en este proyecto con la consecución del OBJ. 4.

En cuanto a la enfermedad de Duchenne, del total de variaciones extraídas, el algoritmo rechazó casi dos tercios del total, 1766 variaciones (63%). Dentro de las variaciones aceptadas para ser cargadas posteriormente en Delfos, 757 se aceptaron con evidencia limitada (29.3%), 43 con evidencia moderada (1.7%), 15 fueron declaradas como variaciones para mantener en observación hasta que se aportase más información por parte de la comunidad científica (0.6%) y 0 aceptadas con evidencia fuerte (0%).

Con respecto a la enfermedad de Becker, las variaciones rechazadas ascendían a la mitad de las variaciones totales, 61 variaciones de 122. De entre las aceptadas, 40 lo estaban con evidencia limitada (32.8%), 14 variaciones para revisar en un futuro en base a los avances de la comunidad científica (11.5%), 7 aceptadas con evidencia moderada (5.7%) y 0 aceptadas con evidencia fuerte (0%).

La mayoría de las variaciones fueron aceptadas con evidencia limitada, lo cual significa que estas no presentan una significancia clínica establecida de acuerdo con los estándares de la ACMG/AMP 2015 y que, además, los estudios que las sustentan no son relevantes según el algoritmo. Esto no quiere decir que no tengan validez, sino que no cumplen los requisitos que establece el algoritmo, que son provenir de estudios GWAS o de otros estudios con un número de participantes > 500 individuos.

Estos requisitos para ser clasificadas con evidencia fuerte son inabarcables dadas las características de las enfermedades analizadas en este trabajo, las cuales se enmarcan dentro del grupo de “enfermedades raras”. La baja prevalencia que presentan en la población (1-9/100000) impide llevar a cabo estudios tipo GWAS, e incluso si consiguen llevarse a cabo [como el estudio de 2018 realizado por Weiss y colaboradores (2018)], estos carecen de la potencia estadística que presentan otros estudios con números mayores de participantes.

Para cerrar este apartado de discusión de los resultados conseguidos a partir del planteamiento del OBJ. 2, se pretende hacer hincapié en la necesidad de conocer las bases de las enfermedades que se analizan mediante este tipo de algoritmos de IA. Del mismo modo, es de vital importancia acompañar a estos resultados de las explicaciones necesarias para que el personal clínico e investigador pueda tomar decisiones en base a una serie de argumentos respaldados científicamente.

5.3. RESULTADOS Y DISCUSIÓN DEL OBJETIVO 3

El OBJ. 3 consistía en evaluar la importancia de la BBDD de LOVD no incluida en la plataforma OraGenDel. Este objetivo se ha alcanzado mediante la realización de una serie de experimentos explicados tanto en el **Subapartado 4.2.3** como en el **4.4.2**.

Los resultados derivados de la validación (**4.2.3**), los cuales se comentan también en el **Subapartado 4.4.2**, fueron un 96.72% de cobertura para BMD y un 99.75% para DMD, siendo las variaciones en común obtenidas para la enfermedad de Becker 59/61 y, para la enfermedad de Duchenne, 810/812. Con estos resultados, se pudo llevar a cabo el análisis consistente en determinar si la BBDD de LOVD aportaba información que no estuviese presente en las BBDD de ClinVar o de Ensembl, y viceversa. Los resultados obtenidos se muestran en la tabla siguiente (**Tabla 6**).

Tabla 6. Valores de cobertura entre la BBDD de LOVD con respecto a Ensembl y ClinVar, y viceversa.

	LOVD / Delfos*		Delfos* / LOVD	
	En común / Total	%	En común / Total	%
Duchenne (DMD)	264/**812**	32.51	264/**5392**	4.90
Becker (BMD)	22/**61**	36.07	22/**5392**	0.41

* Delfos = Ensembl + ClinVar

Los resultados que se observan en la tabla hacen referencia a la misma comparación vista desde dos perspectivas distintas. Tanto la primera columna como la segunda, muestran el mismo número de variaciones en común, pero la diferencia radica en el cálculo del porcentaje. Mientras que, en la primera columna, se calcula la cobertura de las variaciones en común frente al total de variaciones almacenadas en el repositorio Delfos del OraGenDel (cuyo origen es Ensembl y ClinVar); en la segunda columna, esta se calcula frente al total de variaciones extraídas de LOVD que han sido aceptadas en base a su significancia clínica.

Centrándose en cualquiera de las dos columnas, para la DMD, las variaciones en común entre la BBDD de LOVD y el repositorio de Delfos son 264. En lo referente a la BMD, únicamente 22 variaciones. Estos resultados indican que existen variaciones relevantes que están en Delfos pero no en LOVD y que, del mismo modo, existen variaciones que se encuentran en LOVD pero no en Delfos.

Las razones que pueden llevar a que se den ratios de cobertura tan bajos entre dos repositorios de variaciones genéticas son varias. La primera de ellas, está relacionada con el actual problema de heterogeneidad de las BBDD existentes, que conduce a un incremento en la dispersión de los datos y contribuye a dificultar la tarea de extraer información relevante de manera rápida y efectiva.

La segunda razón, está enfocada a la calidad de las variaciones incluidas en los distintos repositorios. Como se ha explicado en el **Apartado 3.2**, no todas las BBDD se encargan de curar las variaciones que se almacenan en ellas. Concretamente, con el repositorio de LOVD ocurre que, pese a que existe una persona encargada de realizar la tarea de curado de las variaciones (denominada "*curator*"), no siempre se cumple. La manera de saberlo es basándose en la primera columna de la tabla que en la que se observan las variaciones de la BBDD. Esta columna muestra la significancia clínica aportada por la persona que ha cargado la variación (o "*submitter*") y la significancia clínica que le confiere el *curator*. Cuando este último no ha aportado su valoración, se indica con un punto: {"."}. Tras analizar esto último, se observó que un 77.57% de las variaciones almacenadas en dicha BBDD no presentaban la valoración del *curator*.

La tercera, y última, razón que puede llevar a la obtención de estos resultados está relacionada con los *scripts* elaborados con tal fin. Si bien durante la consecución de este objetivo se han realizado pruebas previas para corroborar la validez de dichos *scripts*, no puede descartarse que se haya podido producir algún error. No obstante, en caso de haber sido esta la causa, lo más probable es que los resultados no hubieran sido tan dispares como los obtenidos. Por tanto, con todo lo explicado anteriormente, ante la pregunta de si es imprescindible la BBDD de LOVD para la extracción de variaciones relevantes asociadas a DMD y BMD, la respuesta es que sí, en base a los resultados obtenidos. De esta afirmación, deriva la necesidad de establecer conectores con esta BBDD, lo cual se comenta dentro del **Apartado 5.5** de Propuestas de Mejora.

5.4. RESULTADOS Y DISCUSIÓN DEL OBJETIVO 4

El último objetivo se centraba en la realización de un análisis de las variaciones relevantes que diferenciasen DMD de BMD, según el tipo de cambio y la localización. La manera de proceder para la consecución de este objetivo se explica en el **Subapartado 4.4.3**.

5.4.1. Según el tipo de cambio

Los resultados obtenidos de dicho análisis fueron los que se muestran en las tablas siguientes (**Tabla 7** y **Tabla 8**).

Tabla 7. Clasificación de las variaciones de Duchenne según el tipo de cambio.

Duchenne (DMD)		**Delphos (E+C)**	
Nº variaciones		812	
Cambio	SNV	378	46.55 %
	Nonsense	269	33.01 % (70.90 %)
	Missense	112	13.55 % (29.10 %)
	Deleción	294	36.21 %
	Duplicación	104	12.80 %
	Microsatélites	27	3.33 %
	Indel	3	0.37 %
	Pérdida núm. copias	3	0.37 %
	Inserción	3	0.37 %
Cambio pauta lectura		199	24.51 %

La **Tabla 7** muestra una clasificación de las variaciones asociadas a Duchenne que se encuentran en el repositorio de datos genómicos de Delfos. Para la DMD, la plataforma acepta un total de 812 variaciones, de entre las cuales la mayoría se corresponden con SNVs (46.55%) y deleciones (36.21%). Además, dentro de las SNVs, la mutaciones más frecuentes son las de tipo *nonsense*, que producen un codón de parada prematuro. Por último, cabe destacar también que casi un cuarto del total de las variaciones (24.51%) alteran la pauta de lectura de la maquinaria de traducción, lo que contribuye a la imposibilidad de obtener una proteína funcional.

Estos porcentajes se corresponden con las aportaciones de los expertos en distrofinopatías del IIS La Fe de Valencia y con lo explicado previamente en el **Subapartado 1.2.1**, dado que las personas que padecen la enfermedad de Duchenne suelen presentar una ausencia de distrofina en el tejido muscular debido a este tipo de alteraciones genéticas.

Tabla 8. Clasificación de las variaciones de Becker según el tipo de cambio.

Becker (BMD)		**Delphos (E+C)**	
Nº variaciones		61	
Cambio	SNV	52	85.25 %
	Nonsense	49	80.33 % (94.23 %)
	Missense	3	4.92 % (5.77 %)
	Deleción	4	6.56 %
	Duplicación	1	1.64 %
	Indel	1	1.64 %
Cambio pauta lectura		3	4.92 %

En lo mostrado en la **Tabla 8** referente a la BMD, de las 61 variaciones aceptadas por la plataforma OraGenDel que se asocian a esta enfermedad, la mayoría se corresponden con SNVs. En concreto, dentro de la subclasificación de estas últimas, la mayor parte de las mutaciones se tratan de mutaciones *nonsense*, al igual que ocurría con la DMD. En cuanto a las mutaciones que alteran la pauta de lectura, se observa un porcentaje claramente inferior al mostrado en la **Tabla 3**, de un 4.9% con respecto al 24.51% que presentaban las variaciones asociadas a la DMD.

Estos resultados, al contrario que los anteriores, no se correspondían del todo con las aportaciones de los expertos. El hecho de que hubiesen únicamente 3 variaciones que alteraban la pauta de lectura sí que se adecuaba a la hipótesis asociada a la enfermedad de Becker; pero que la mayoría de SNVs fuesen de tipo *nonsense*, no. Esta hipótesis consiste en que las variaciones que conducen al desarrollo de la forma más atenuada de estas distrofinopatías han de ser de tipo *missense*, dado que estos pacientes suelen llegar a albergar una pequeña cantidad de distrofina funcional, cuya cantidad depende del tipo de cambio aminoacídico que se produzca. Según esta hipótesis, las SNVs de tipo *missense* abarcarían ~70% del total de variaciones asociadas a la distrofia muscular de Becker.

Con tal de dilucidar la causa de esta discrepancia, se analizó qué variaciones tenían en común los fenotipos de DMD y de BMD, y lo que se vio fue que un total de 58 variaciones se encontraban tanto asociadas a la DMD como a la BMD, como se muestra en la **Figura 9**. En el caso de la DMD, esto supone un 7.14% del total de variaciones (58/812); mientras que para la BMD, el 95.08% (58/61). Con este resultado, se deduce que, al estar incluida en el fenotipo de DMD prácticamente la totalidad de las variaciones que se asocian también a la BMD, no es de extrañar que ambas hayan obtenido resultados similares tras la clasificación. No obstante, este hecho no llega a justificar por qué no se dio el caso de que las variaciones en común fuesen precisamente parte de las 112 SNVs de tipo *missense* que se asociaban a Duchenne.

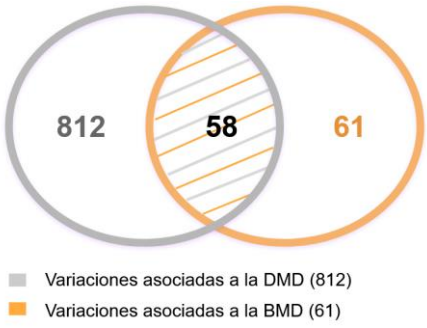

Figura 9. Número de variaciones asociadas a la DMD, a la BMD y en común para ambas.

Las últimas opciones capaces de explicar el resultado de la **Tabla 8** son tres. La primera es que se requieren más estudios centrados en la BMD con tal de confirmar las hipótesis acerca de los mecanismos moleculares de la misma. La segunda, es que existan esas variaciones de tipo *missense* asociadas a Becker en las BBDD, pero que el algoritmo las haya descartado por no cumplir con los requisitos establecidos por el mismo. Y, la tercera, es que en la BBDD de LOVD se encuentren más variaciones de tipo *missense* que se asocien a la BMD. Para saber esto último, dado que este repositorio no hace distinción entre fenotipos (únicamente indica que las variaciones se encuentran en el gen *DMD*), se tendría que realizar un estudio detallado de las publicaciones asociadas a cada variación de tipo *missense* presentes en esta BBDD. Dicho análisis queda a la disposición de futuras investigaciones a realizar en este ámbito.

5.4.2. Según la localización

Con respecto al análisis de las variaciones de la DMD y de la BMD en base a la localización que ocupan en el gen *DMD*, los resultados obtenidos se pueden observar en las gráficas de la **Figura 10** y la **Figura 11**. El eje de abscisas representa el número de exones (y de sus respectivas secuencias intrónicas adyacentes) que contiene el gen *DMD*. Por otro lado, el eje de ordenadas muestra el número de variaciones que se encuentran en una determinada localización.

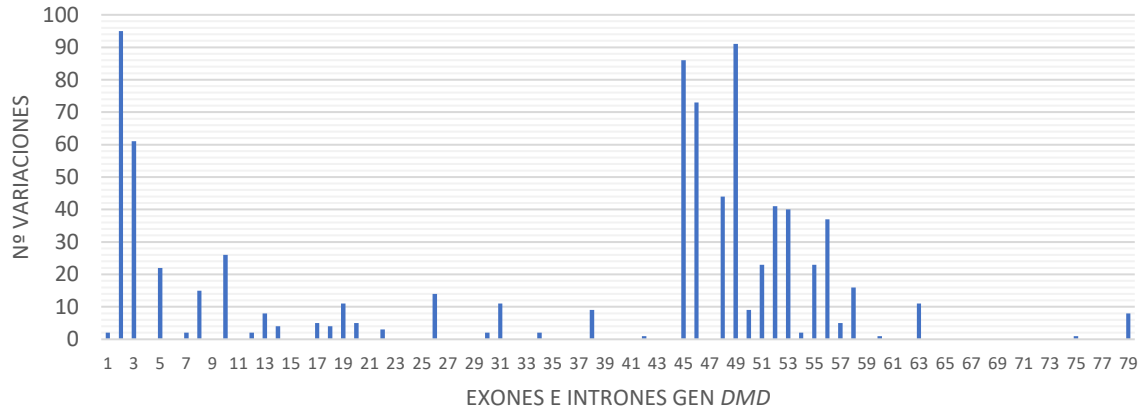

Figura 10. Localización de las 812 variaciones obtenidas por OraGenDel para Duchenne en el gen *DMD*.

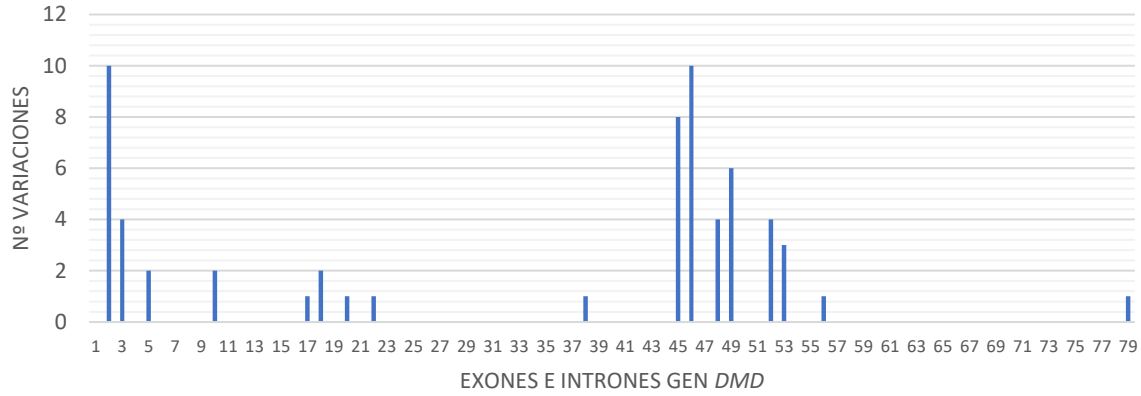

Figura 11. Localización de las 61 variaciones obtenidas por OraGenDel para Becker en el gen *DMD*.

Tanto en la primera gráfica como en la segunda, se observa, a grandes rasgos, un patrón similar. Lo primero que destaca es la presencia de dos *hot-spots* o sitios con propensión a acumular mutaciones. El primero se encuentra en los exones 2 y 3. El segundo abarca desde el exón 45 hasta el 57. Además, otro hecho relevante es la presencia de pequeños núcleos mutacionales dispersos entre el primer *hot-spot* y el segundo. Por último, se contabiliza también un reducido número de mutaciones localizadas en la parte final del gen, en la zona del exón 79.

Ahora bien, si se pretende llevar a cabo un análisis de las diferencias que presentan ambos fenotipos, únicamente se obtienen ciertas regiones para las cuales los pacientes de DMD presentan alteraciones y en las cuales no han sido anotadas variaciones relacionadas con la BMD. Sin embargo, dichas diferencias no pueden establecerse como un resultado completamente fiable, debido a la disparidad en el número de variaciones asociadas a cada fenotipo. Para poderse realizar una

investigación de este tipo, se requeriría de un mayor número de variaciones en el caso de la BMD.

Con tal de extraer significado biológico al hecho de que haya zonas con tendencia a acumular mutaciones, se analizó con qué región de la secuencia polipeptídica de la distrofina se correspondían dichas alteraciones. Tras el análisis, se determinó que el primer *hot-spot* se localiza en la región N-terminal de la proteína, concretamente en el dominio de unión a actina, o ABD. El segundo *hot-spot*, en cambio, se corresponde con la zona del rodo dominio central, concretamente, desde la repetición 17 (R17) de espectrina hasta la 22 (R22). Estas repeticiones son precisamente las que albergan dominios de unión con las subunidades que conforman el dímero de sintrofina, así como también con los microtúbulos del citoesqueleto.

Lo que se deduce a partir de lo comentado anteriormente es que las mutaciones, tanto de Duchenne como de Becker, suelen producirse en aquellas zonas que presentan interacciones con otros componentes, ya sean del complejo DAPC o constituyentes del resto de estructuras celulares.

Un hecho a destacar también a partir de los resultados y las conclusiones extraídas para la consecución de este objetivo es que, tanto los datos mostrados en las gráficas como las discusiones aportadas, coincidieron con las contribuciones de los expertos, lo cual respalda los resultados obtenidos por la plataforma OraGenDel, a la vez que confiere las garantías necesarias para confiar en dichos resultados.

5.5. PROPUESTAS DE MEJORA

En este apartado, se comentan las propuestas de mejora enfocadas a impulsar el funcionamiento de la plataforma OraGenDel, tal y como establece el último paso del método empleado en el presente proyecto (**Apartado 4.5**).

La primera propuesta de mejora consiste en establecer conectores desde la plataforma OraGenDel con la BBDD de LOVD. En base a la discusión de los resultados del OBJ. 3, realizada en el **Subapartado 5.3**, si se pretende obtener la totalidad de las variaciones relevantes asociadas a las enfermedades de DMD y BMD, es necesario disponer de una parte de las variaciones que alberga este repositorio. En este trabajo, se ha contribuido a la consecución de esta mejora aportando una serie de *scripts* que han sido creados con tal fin y que se explican en el **Anexo 1**.

La segunda propuesta está relacionada con el formato del nombre de las variaciones que proporciona OraGenDel (explicado en detalle en el **Subapartado 4.2.2**). Durante el filtrado de estas últimas para la eliminación de redundancias, se observó que había variaciones que eran eliminadas por tener el mismo nombre, pero que cuando se analizaban en detalle, presentaban un tipo de cambio diferente. En concreto, las que daban lugar a problemas eran aquellas con cambios mayores de 10 nucleótidos, ya que el cambio asociado a las mismas dejaba de estar en formato *'{ref | ?} > {alt | ?}'* para pasar a poner únicamente *'{longchange}'*. La propuesta de mejora en este caso, consiste en indicar el tipo de cambio, cuando este abarque más de 10 nucleótidos, de la siguiente manera: *'{longchange_type}'*, donde "*type*" se sustituiría por "{del | ins | dup | inv | indel}". De esta manera, se solucionaría el problema y se estaría contribuyendo a una mejor precisión a la hora de establecer el formato del nombre.

6. CONCLUSIONES Y LÍNEAS FUTURAS

Con la realización del presente proyecto final de grado, se ha contribuido a la mejora de la gestión de datos genómicos y al estudio de las enfermedades distrofia muscular de Duchenne y distrofia muscular de Becker.

La importancia de profundizar en el conocimiento de dichas patologías, enmarcadas dentro del contexto de las Enfermedades Raras, radica en descubrir las causas que conducen a su aparición, con tal de diseñar estrategias encaminadas a mejorar la vida de los pacientes y a encontrar terapias cada vez más efectivas. Además, cualquier aportación, por pequeña que sea, que dé soporte a visibilizar y comprender estas enfermedades contribuye, de alguna manera, a que estas dejen de ser las grandes olvidadas. En este sentido, este trabajo ha abierto una nueva vía para el estudio de las mismas, desde un punto de vista basado en el empleo de técnicas bioinformáticas enmarcadas dentro del contexto de la IAE.

Con respecto a la mejora en la gestión de datos genómicos, se ha validado el funcionamiento de la plataforma Oráculo Genómico de Delfos con éxito y se han propuesto mejoras para permitir el refinamiento de la misma. Además, el hecho de haber desarrollado *scripts* en un lenguaje de programación distinto al empleado por la plataforma (como se explica en el **Apartado 3.1**), ha permitido el surgimiento de una cuestión que no se había valorado hasta la fecha. Esta cuestión consiste en el planteamiento de ampliar la arquitectura *software* de la plataforma OraGenDel con tal de aceptar la inclusión de código proveniente de otros lenguajes de programación. Con esto, se ha contribuido a dirigir la evolución de la plataforma hacia su universalización, lo que le confiere una potencialidad mayor de la que ya albergaba *per se*.

Por último, en cuanto a las futuras líneas de investigación que se plantean abordar tras la realización de este proyecto, en el **Subapartado 5.4.1** surge la necesidad de realizar un futuro estudio de las variaciones de tipo *missense* de la BBDD de LOVD con el fin de determinar cuáles del total han sido anotadas a partir de estudios realizados para la enfermedad de Becker. Ahora bien, en términos generales, existen más líneas futuras de investigación. Algunas de estas últimas van enfocadas a la realización de más validaciones de la plataforma OraGenDel pero centrándose, esta vez, en la extracción de variaciones relevantes asociadas a otras enfermedades como, por ejemplo, las distrofias de retina.

7. REFERENCIAS BIBLIOGRÁFICAS

AMOASII, L.; HILDYARD, J. C. W.; LI, H.; SANCHEZ-ORTIZ, E.; MIREAULT, A.; CABALLERO, D.; HARRON, R.; STATHOPULOU, T.-R.; MASSEY, C.; SHELTON, J. M.; BASSEL-DUBY, R.; PIERCY, R. J. y OLSON, E. N. (2018). Gene editing restores dystrophin expression in a canine model of Duchenne muscular dystrophy. *Science*, *362*(6410), 86-91.

AYALON, G.; DAVIS, J. Q.; SCOTLAND, P. B. y BENNETT, V. (2008). An ankyrin-based mechanism for functional organization of dystrophin and dystroglycan. *Cell*, *135*(7), 1189-1200.

BÄCKMAN, E. y HENRIKSSON, K. G. (1995). Low-dose prednisolone treatment in Duchenne and Becker muscular dystrophy. *Neuromuscular Disorders: NMD*, *5*(3), 233-241.

BARREDO ARRIETA, A.; DÍAZ-RODRÍGUEZ, N.; DEL SER, J.; BENNETOT, A.; TABIK, S.; BARBADO, A.; GARCÍA, S.; GIL-LÓPEZ, S.; MOLINA, D., BENJAMINS, R.; CHATILA, R. y HERRERA, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82-115.

BELLO, L.; CAMPADELLO, P.; BARP, A.; FANIN, M.; SEMPLICINI, C.; SORARÙ, G.; CAUMO, L.; CALORE, C.; ANGELINI, C. y PEGORARO, E. (2016). Functional changes in Becker muscular dystrophy: Implications for clinical trials in dystrophinopathies. *Scientific Reports*, *6*(1), 32439.

BUNIELLO, A.; MacARTHUR, J. A. L.; CEREZO, M.; HARRIS, L. W.; HAYHURST, J.; MALANGONE, C.; McMAHON, A.; MORALES, J.; MOUNTJOY, E.; SOLLIS, E.; SUVEGES, D.; VROUSGOU, O.; WHETZEL, P. L.; AMODE, R.; GUILLEN, J. A.; RIAT, H. S.; TREVANION, S. J.; HALL, P.; JUNKINS, H.; FLICEK, P.; BURDETT, T.; HINDORFF, L. A.; CUNNINGHAM, F. y PARKINSON, H. (2019). The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019. *Nucleic Acids Research*, *47*(D1), D1005-D1012.

CASKEY, C. T.; NUSSBAUM, R. L.; COHAN, L. C. y POLLACK, L. (1980). Sporadic occurrence of Duchenne muscular dystrophy: Evidence for new mutation. *Clinical Genetics*, *18*(5), 329-341.

CHELLY, J.; HAMARD, G.; KOULAKOFF, A.; KAPLAN, J.-C.; KAHN, A. y BERWALD-NETTER, Y. (1990). Dystrophin gene transcribed from different promoters in neuronal and glial cells. *Nature*, *344*(6261), 64-65.

COSTA M.; LEÓN A. y PASTOR Ó. (2020) The Importance of the Temporal Dimension in Identifying Relevant Genomic Variants: A Case Study. En: Grossmann G., Ram S. (eds) *Advances in Conceptual Modeling. ER 2020. Lecture Notes in Computer Science*, vol 12584. Springer, Cham.

D'SOUZA, V. N.; NGUYEN, T. M.; MORRIS, G. E.; KARGES, W., PILLERS, D. A. y RAY, P. N. (1995). A novel dystrophin isoform is required for normal retinal electrophysiology. *Human Molecular Genetics*, *4*(5), 837-842.

DUAN, D. (2018). Micro-Dystrophin Gene Therapy Goes Systemic in Duchenne Muscular Dystrophy Patients. *Human Gene Therapy*, *29*(7), 733-736.

DUAN, D.; GOEMANS, N.; TAKEDA, S.; MERCURI, E. y AARTSMA-RUS, A. (2021). Duchenne muscular dystrophy. *Nature Reviews Disease Primers*, *7*(1), 1-19.

FERNÁNDEZ-SUÁREZ, X. M.; RIDGEN, D. J. y GALPERIN, M. Y. (2014). The 2014 Nucleic Acids Research Database Issue and an updated NAR online Molecular Biology Database Collection. *Nucleic Acids Research*, *42*(Database issue), D1-6.

FOKKEMA, I. F. A. C.; TASCHNER, P. E. M.; SCHAAFSMA, G. C. P.; CELLI, J.; LAROS, J. F. J. y DEN DUNNEN, J. T. (2011). LOVD v.2.0: The next generation in gene variant databases. *Human Mutation*, *32*(5), 557-563.

FRATTER, C.; DALGEISH, R.; ALLEN, S. K.; SANTOS, R.; ABBS, S.; TUFFERY-GIRAUD, S. y FERLINI, A. (2020). EMQN best practice guidelines for genetic testing in dystrophinopathies. *European Journal of Human Genetics*, *28*(9), 1141-1159.

GAO, Q.Q. y MCNALLY, E.M. (2015). The Dystrophin Complex: Structure, Function, and Implications for Therapy. In *Comprehensive Physiology*, R. Terjung (Ed.).

GARCÍA S., A.; PALACIO, A. L.; REYES ROMÁN, J. F.; CASAMAYOR, J. C. y PASTOR, O. (2020). Towards the Understanding of the Human Genome: A Holistic Conceptual Modeling Approach. *IEEE Access*, *8*, 197111-197123.

GARCÍA, S.; DE HARO, T.; ZAFRA-CERES, M.; POYATOS, A.; GOMEZ-CAPILLA, J. A. y GOMEZ-LLORENTE, C. (2014). Identification of de novo Mutations of Duchenne/Becker Muscular Dystrophies in Southern Spain. *International Journal of Medical Sciences*, *11*(10), 988-993.

GÓRECKI, D. C.; MONACO, A. P.; DERRY, J. M.; WALKER, A. P.; BARNARD, E. A. y BARNARD, P. J. (1992). Expression of four alternative dystrophin transcripts in brain regions regulated by different promoters. *Human Molecular Genetics*, *1*(7), 505-510.

HAAS, M.; VLCEK, V.; BALABANOV, P.; SALMONSON, T.; BAKCHINE, S.; MARKEY, G.; WEISE, M.; SCHLOSSER-WEBER, G.; BROHMANN, H.; YERRO, C. P.; MENDIZABAL, M. R.; STOYANOVA-BENINSKA, V. y HILLEGE, H. L. (2015). European Medicines Agency review of ataluren for the treatment of ambulant patients aged 5 years and older with Duchenne muscular dystrophy resulting from a nonsense mutation in the dystrophin gene. *Neuromuscular Disorders: NMD*, *25*(1), 5-13.

HO, D.; QUAKE, S. R.; MCCABE, E. R. B.; CHNG, W. J.; CHOW, E. K.; DING, X.; GELB, B. D.; GINSBURG, G. S.; HASSENSTAB, J.; HO, C.-M.; MOBLEY, W. C.; NOLAN, G. P.; ROSEN, S. T.; TAN, P.; YEN, Y. y ZARRINPAR, A. (2020). Enabling Technologies for Personalized and Precision Medicine. *Trends in Biotechnology*, *38*(5), 497-518.

HOFFMAN, E. P.; BROWN, R. H. y KUNKEL, L. M. (1987). Dystrophin: The protein product of the Duchenne muscular dystrophy locus. *Cell*, *51*(6), 919-928.

HOWE, K. L.; ACHUTHAN, P.; ALLEN, J.; ALVAREZ-JARRETA, J.; AMODE, M. R.; ARMEAN, I. M.; AZOV, A. G.; BENNETT, R.; BHAI, J.; BILLIS, K.; BODDU, S.; CHARKHCHI, M.; CUMMINS, C.; DA RIN FIORETTO, L.; DAVIDSON, C.; DODIYA, K., EL HOUDAIGUI, B.; FATIMA, R.; GALL, A.; GIRON, C. G.; GREGO, T.; GUIJARRO-CLARKE, C.; HAGGERTY, L.; HEMROM, A.; HOURLIER, T.; IZUOGU, O. G.; JUETTEMANN, T.; KAIKALA, V.; KAY, M.; LAVIDAS, I.; LE, T.; LEMOS, D.; MARTINEZ, J. G.; MARUGÁN, J. C.; MAUREL, T.; MCMAHON, A. C.; MOHANAN, S.; MOORE, B.; MUFFATO, M.; OHEH, D. N.; PARASCHAS, D.; PARKER, A.; PARTON, A.; PROSOVETSKAIA, I.; SAKTHIVEL, M. P.; SALAM, A. A.; SCHMITT, B. M.; SCHUILENBURG, H.; SHEPPARD, D.; STEED, E.; SZPAK, M.; SZUBA, M.; TAYLOR, K.; THORMANN, A.; THREADGOLD, G.; WALTS, B.; WINTERBOTTOM, A.; CHAKIACHVILI, M.; CHAUBAL, A.; DE SILVA, N.; FLINT, B.; FRANKISH, A.; HUNT, S. E.; IISLEY, G. R.; LANGRIDGE, N.; LOVELAND, J. E.; MARTIN, F. J.; MUDGE, J. M.; MORALES, J.; PERRY, E.; RUFFIER, M.; TATE, J.; THYBERT, D.; TREVANION, S. J.; CUNNINGHAM, F.; YATES, A. D.; ZERBINO, D. R. y FLICEK, P. (2021). Ensembl 2021. *Nucleic Acids Research*, *49*(D1), D884-D891.

HUGNOT, J. P.; GILGENKRANTZ, H.; VINCENT, N.; CHAFEY, P.; MORRIS, G. E.; MONACO, A. P.; BERWALD-NETTER, Y., KOULAKOFF, A.; KAPLAN, J. C. y KAHN, A. (1992). Distal transcript of the dystrophin gene initiated from an alternative first exon and encoding a 75-kDa protein widely distributed in nonmuscle tissues. *Proceedings of the National Academy of Sciences of the United States of America*, *89*(16), 7506–7510.

HULSEN, T.; JAMUAR, S. S.; MOODY, A. R.; KARNES, J. H.; VARGA, O.; HEDENSTED, S.; SPREAFICO, R.; HAFLER, D. A. y MCKINNEY, E. F. (2019). From Big Data to Precision Medicine. *Frontiers in Medicine*, *6*.

HUNT, S. E.; MCLAREN, W.; GIL, L.; THORMANN, A.; SCHUILENBURG, H.; SHEPPARD, D.; PARTON, A.; ARMEAN, I. M.; TREVANION, S. J.; FLICEK, P. y CUNNINGHAM, F. (2018). Ensembl variation resources. *Database*, *2018*(bay119).

HUTCHINGS, J. R. A. (2020). Chapter 4—Genomic databases. En D. A. Forero & G. P. Patrinos (Eds.), *Genome Plasticity in Health and Disease* (pp. 47-62). Academic Press.

IBRAGHIMOV-BESKROVNAYA, O.; ERBASTI, J. M.; LEVEILLE, C. J.; SLAUGHTER, C. A.; SERNETT, S. W. y CAMPBELL, K. P. (1992). Primary structure of dystrophin-associated glycoproteins linking dystrophin to the extracellular matrix. *Nature*, *355*(6362), 696-702.

JOSHI, J. (2021). Chapter 9 – Python, a reliable programming language for chemoinformatics and bioinformatics. En N. Sharma, H. Ojha, P. K. Raghav, & R. k. Goyal (Eds.), *Chemoinformatics and Bioinformatics in the Pharmaceutical Sciences* (pp. 279-304). Academic Press.

KOENIG, M.; BEGGS, A. H.; MOYER, M.; SCHERPF, S.; HEINDRICH, K., BETTECKEN, T.; MENG, G.; MÜLLER, C. R.; LINDLÖF, M.; KAARIAINEN, H.; DE LA CHAPELLET, A.; KIURU, A.; SAVONTAUS, M. L.; GILGENKRANTZ, H.; RÉCAN, D.; CHELLY, J.; KAPLAN, J. C.; COVONE, A. E.; ARCHIDIACONO, N.; ROMEO, G.; LIECHTI-GAILATI, S.; SCHNEIDER, V.; BRAGA, S.; MOSER, H.; DARRAS, B. T.; MURPHY, P.; FRANCKE, U.; CHEN, J. D.; MORGAN, G.; DENTON, M.; GREENBERG, C. R.; WROGEMANN, K.; BLONDEN, L. A.; VAN PAASSEN, M. B.; VAN OMMEN, G. J. y KUNKEL, L. M. (1989). The molecular basis for Duchenne versus Becker muscular dystrophy: Correlation of severity with type of deletion. *American Journal of Human Genetics*, *45*(4), 498-506.

KULKARNI, S.; SENEVIRATNE, N.; BAIG, M. S. y KHAN, A. H. A. (2020). Artificial Intelligence in Medicine: Where Are We Now? *Academic Radiology*, *27*(1), 62-70.

LANDRUM, M. J.; LEE, J. M.; BENSON, M.; BROWN, G. R.; CHAO, C.; CHITIPIRALLA, S.; GU, B.; HART, J.; HOFFMAN, D.; JANG, W.; KARAPETYAN, K.; KATZ, K.; LIU, C.; MADDIPATLA, Z.; MALHEIRO, A.; MCDANIEL, K.; OVETKY, M.; RILEY, G.; ZHOU, G.; HOLMES, J. B.; KATTMAN, B. L. y MAGLOTT, D. R. (2018). ClinVar: Improving access to variant interpretations and supporting evidence. *Nucleic Acids Research*, *46*(D1), D1062-D1067.

LEÓN PALACIO A.; GARCÍA GIMÉNEZ A.; CASAMAYOR RÓDENAS J.C. y REYES ROMÁN J.F. (2018) Genomic Data Management in Big Data Environments: The Colorectal Cancer Case. En: Woo C., Lu J., Li Z., Ling T., Li G., Lee M. (eds) *Advances in Conceptual Modeling. ER 2018. Lecture Notes in Computer Science*, vol 11158. Springer, Cham.

LEÓN PALACIO, A. (2019). *SILE: A Method for the Efficient Management of Smart Genomic Information*. Tesis doctoral en Ciencias de la Computación. Universitat Politècnica de València.

LIDOV, H. G.; SELIG, S. y KUNKEL, L. M. (1995). Dp140: A novel 140 kDa CNS transcript from the dystrophin locus. *Human Molecular Genetics*, *4*(3), 329-335.

LIM, B. C.; LEE, S.; SHIN, J.-Y.; KIM, J.-I.; HWANG, H.; KIM, K. J.; HWANG, Y. S.; SEO, J.-S. y CHAE, J. H. (2011). Genetic diagnosis of Duchenne and Becker muscular dystrophy using next-generation sequencing technology: Comprehensive mutational search in a single platform. *Journal of Medical Genetics*, *48*(11), 731-736.

ŁOBODA, A. y DULAK, J. (2020). Muscle and cardiac therapeutic strategies for Duchenne muscular dystrophy: Past, present, and future. *Pharmacological Reports*, *72*(5), 1227-1263.

LUCE, L.; CARCIONE, M.; MAZZANTI, C., BUONFIGLIO, P. I.; DALAMÓN, V.; MESA, L.; DUBROVSKY, A.; CORDERÍ, J. y GILIBERTO, F. (2021). Theragnosis for Duchenne Muscular Dystrophy. *Frontiers in Pharmacology*, *12*.

MENDELL, J. R.; SAHENK, Z.; MALIK, V.; GOMEZ, A. M.; FLANIGAN, K. M.; LOWES, L. P.; ALFANO, L. N.; BERRY, K.; MEADOWS, E.; LEWIS, S.; BRAUN, L.; SHONTZ, K.; ROUHANA, M.; CLARK, K. R.; ROSALES, X. Q.; AL-ZAIDY, S.; GOVONI, A.; RODINO-KLAPAC, L. R.; HOGAN, M. J. y KASPAR, B. K. (2015). A phase 1/2a follistatin gene therapy trial for becker muscular dystrophy. *Molecular Therapy: The Journal of the American Society of Gene Therapy*, *23*(1), 192-201.

MOXLEY, R. T.; ASHWAL, S.; PANDYA, S.; CONNOLLY, A.; FLORENCE, J.; MATHEWS, K.; BAUMBACH, L.; MCDONALD, C.; SUSSMAN, M. y WADE, C. (2005). Practice Parameter: Corticosteroid treatment of Duchenne dystrophy: Report of the Quality Standards Subcommittee of the American Academy of Neurology and the Practice Committee of the Child Neurology Society. *Neurology*, *64*(1), 13-20.

NAKAMURA, A. (2015). X-Linked Dilated Cardiomyopathy: A Cardiospecific Phenotype of Dystrophinopathy. *Pharmaceuticals*, *8*(2), 303-320.

NAKAMURA, A.; FUEKI, N.; SHIBA, N.; MOTOKI, H.; MIYAZAKI, D.; NISHIZAWA, H.; ECHIGOYA, Y.; YOKOTA, T.; AOKI, Y. y TAKEDA, S. (2016). Deletion of exons 3−9 encompassing a mutational hot spot in the DMD gene presents an asymptomatic phenotype, indicating a target region for multiexon skipping therapy. *Journal of Human Genetics*, *61*(7), 663-667.

NASCIMENTO OSORIO, A.; MEDINA CANTILLO, J.; CAMACHO SALAS, A.; MADRUGA GARRIDO, M. y VILCHEZ PADILLA, J. J. (2019). Consenso para el diagnóstico, tratamiento y seguimiento del paciente con distrofia muscular de Duchenne. *Neurología*, *34*(7), 469-481.

OMIM - Online Mendelian Inheritance in Man. (s. f.). *Muscular Dystrophy, Duchenne Type; DMD.* Recuperado 12 de mayo de 2021, de https://omim.org/entry/310200

OMIM - Online Mendelian Inheritance in Man. (s. f.). *Muscular Dystrophy, Becker Type; BMD.* Recuperado 12 de mayo de 2021, de https://omim.org/entry/300376

ORPHANET (2020). *Distrofia muscular de Duchenne*. Recuperado 25 de junio de 2021, de https://www.orpha.net/consor/www/cgi-bin/OC_Exp.php?lng=ES&Expert=98896

PALACIO, A. L. y LÓPEZ, Ó. P. (2018). Smart Data for Genomic Information Systems: The SILE Method. *Complex Systems Informatics and Modeling Quarterly*, *0*(17), 1-23.

PASTOR, O. (2008). Conceptual Modeling Meets the Human Genome. En: Li Q., Spaccapietra S., Yu E., Olivé A. (eds) *Conceptual Modeling. ER 2008*. *Lecture Notes in Computer Science*, vol 5231. Springer, Berlin, Heidelberg.

REYES ROMÁN, JF. (2018). *Diseño y desarrollo de un sistema de información genómica basado en un modelo conceptual holístico del genoma humano.* Tesis Doctoral en Informática. Universitat Politècnica de València.

RICHARDS, S.; AZIZ, N.; BALE, S.; BICK, D.; DAS, S.; GASTIER-FOSTER, J.; GRODY, W. W.; HEGDE, M.; LYON, E.; SPECTOR, E.; VOELKERDING, K. y REHM, H. L. (2015). Standards and guidelines for the interpretation of sequence variants: A joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology. *Genetics in Medicine*, *17*(5), 405-423.

RIGDEN, D. J., y FERNÁNDEZ, X. M. (2021). The 2021 Nucleic Acids Research database issue and the online molecular biology database collection. *Nucleic Acids Research*, *49*(D1), D1-D9.

RYDER, S.; LEADLEY, R. M.; ARMSTRONG, N.; WESTWOOD, M.; DE KOCK, S.; BUTT, T.; JAIN, M. y KLEIJNEN, J. (2017). The burden, epidemiology, costs and treatment for Duchenne muscular dystrophy: An evidence review. *Orphanet Journal of Rare Diseases*, *12*(1), 79.

SCHOUTEN, J. P.; MCELGUNN, C. J.; WAAIJER, R.; ZWIJNENBURG, D.; DIEPVENS, F. y PALS, G. (2002). Relative quantification of 40 nucleic acid sequences by multiplex ligation-dependent probe amplification. *Nucleic Acids Research*, *30*(12), e57.

SHEIKH, O. y YOKOTA, T. (2020). Advances in Genetic Characterization and Genotype–Phenotype Correlation of Duchenne and Becker Muscular Dystrophy in the Personalized Medicine Era. *Journal of Personalized Medicine*, *10*(3), 111.

SOLANKI, V.; ARORA, N.; HASHMI, F. y RAGHUVANSHI, D. (2020). *Computational of Bioinformatics*. 128-131.

SPREEUWENBERG, S. (2019). *Artificial Intelligence needs explanation: Why and how transparency increases the success of AI* solutions. Ed. P. Henao.

SZIGYARTO, C. A.-K. y SPITALI, P. (2018). Biomarkers of Duchenne muscular dystrophy: Current findings. *Degenerative Neurological and Neuromuscular Disease*, *8*, 1-13.

TOH, Z. Y. C.; AUNG-HTUT, M. T.; PINNIGER, G.; ADAMS, A. M.; KRISHNASWARMY, S.; WONG, B. L.; FLETCHER, S. y WILTON, S. D. (2016). Deletion of Dystrophin In-Frame Exon 5 Leads to a Severe Phenotype: Guidance for Exon Skipping Strategies. *PLOS ONE*, *11*(1), e0145620.

TYLER, K. L. (2003). Origins and early descriptions of "Duchenne muscular dystrophy". *Muscle & Nerve*, *28*(4), 402-422.

WAGNER, K. R.; FLECKENSTEIN, J. L.; AMATO, A. A.; BAROHN, R. J.; BUSHBY, K., ESCOLAR, D. M.; FLANINGAN, K. M.; PESTRONK, A.; TAWIL, R.; WOLFE, G. I.; EAGLE, M.; FLORENCE, J. M.; KING, W. M.; PANDYA, S.; STRAUB, V.; JUNEAU, P.; MEYERS, K.; CSIMMA, C.; ARAUJO, T.; ALLEN, R.; PARSONS, S.A.; WOZNEY, J.M.; LAVALLIE, E.R. y MENDELL, J. R. (2008). A phase I/II trial of MYO-029 in adult subjects with muscular dystrophy. *Annals of Neurology*, *63*(5), 561-571.

WAY, M.; POPE, B.; CROSS, R. A.; KENDRICL-JONES, J. y WEEDS, A. G. (1992). Expression of the N-terminal domain of dystrophin in E. coli and demonstration of binding to F-actin. *FEBS Letters*, *301*(3), 243-245.

WEISS, R. B.; VIELAND, V. J.; DUNN, D. M.; KAMINOH, Y. y FLANIGAN, K. M. (2018). Long-range genomic regulators of THBS1 and LTBP4 modify disease severity in Duchenne muscular dystrophy. *Annals of Neurology*, *84*(2), 234-245.

YOTOVA, V.; LEFEBVRE, J.-F.; KOHANY, O.; JURKA, J.; MICHALSKI, R.; MODIANO, D.; UTERMANN, G.; WILLIAMS y S. M.; LABUDA, D. (2007). Tracing genetic history of modern humans using X-chromosome lineages. *Human Genetics*, *122*(5), 431-443.

ZHANG, Y.; LI, H.; MIN, Y.-L.; SANCHEZ-ORTIZ, E.; HUANG, J.; MIREAULT, A. A.; SHELTON, J. M.; KIM, J.; MAMMEN, P. P. A.; BASSEL-DUBY, R. y OLSON, E. N. (2020). Enhanced CRISPR-Cas9 correction of Duchenne muscular dystrophy in mice by a self-complementary AAV delivery system. *Science Advances*, *6*(8), eaay6812.

ZIEMANN, P.; HÖLSCHER, K. y GOGOLLA, M. (2005). Coherently Explaining UML Statechart and Collaboration Diagrams by Graph Transformations. *Electronic Notes in Theoretical Computer Science*, *130*, 263-280.

8. ANEXOS

ANEXO 1. Manual de Usuario para el Empleo de Scripts Desarrollados en Python para la Identificación de Variaciones Genéticas Relevantes

ANEXO 2. MCGH v.3

ANEXO 1

- *Manual de Usuario para el Empleo de Scripts Desarrollados en Python para la Identificación de Variaciones Genéticas Relevantes* -

1. JUSTIFICACIÓN Y PLANTEAMIENTO

Como se comenta a lo largo del desarrollo de la memoria, para el tratamiento de una parte de la información analizada, principalmente de la información extraída de la base de datos de LOVD, se han desarrollado programas o *scripts* escritos en el lenguaje de programación Python. La razón por la que se hizo uso de este lenguaje fue su versatilidad a la hora de analizar conjuntos de datos de naturaleza diversa, así como también por la precisión con la que permitía cumplir con los tipos de análisis que se llevaron a cabo (Joshi, 2021).

Este manual pretende explicar la función que realiza cada *script*, así como mostrar una parte del trabajo desempeñado durante el presente proyecto final de grado, con objeto de proporcionar una justificación fundamentada de los resultados obtenidos. Esto también permitirá que cualquier usuario que pretenda extraer información de variaciones haciendo uso de la base de datos de LOVD y de la plataforma OraGenDel, disponga de la información necesaria para utilizar el soporte arquitectónico de *software* requerido para cumplir este objetivo.

Si bien el objetivo principal de estos *scripts* consistió en analizar las variaciones de la BBDD de LOVD, también se desarrollaron otros que permitieron gestionar la información de las BBDD de ClinVar y Ensembl. El motivo por el cual se crearon *scripts* para extraer una información que ya había sido extraída previamente mediante la plataforma OraGenDel fue validar el funcionamiento de los mismos. Por ello, en este manual se muestran los *scripts* realizados para el manejo de estas tres BBDD (ClinVar, Ensembl y LOVD).

Ahora bien, antes de profundizar en la explicación de los *scripts*, a continuación, se muestra cómo se extrajeron los archivos de las respectivas BBDD para, posteriormente, ser analizados.

2. OBTENCIÓN DE LOS FICHEROS CRUDOS

La extracción manual se realiza por medio de la interfaz web de cada repositorio de datos genómicos. Esta interfaz confiere una serie de herramientas de filtrado que permiten al usuario acceder a la información de manera intuitiva, para posteriormente descargar los resultados de la búsqueda en distintos formatos (p.ej., CSV, TSV, JSON, etc.). Esta forma de acceso está orientada a profesionales clínicos o usuarios en general, para consultas específicas o descargas de cantidades pequeñas de datos (Landrum *et al*., 2018; Howe *et al.*, 2021).

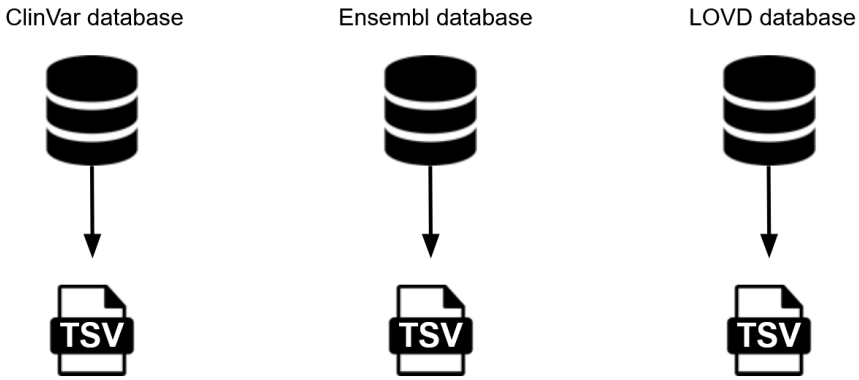

Figura 1A. Representación de la extracción de los ficheros crudos de las BBDD de ClinVar, Ensembl y LOVD.

2.1. ClinVar

En la barra de búsqueda de la interfaz web (*https://www.ncbi.nlm.nih.gov/clinvar/*) se selecciona “Búsqueda avanzada” y se elige la opción “[Disease/Phenotype]” de las opciones del desplegable. Los términos de búsqueda, al ser más de una palabra, se escriben separados por una barra baja: “Duchenne_muscular_dystrophy” y “Becker_muscular_dystrophy”.

A continuación se procede a la descarga del archivo con las variaciones, mediante la opción “*Download*”, que se encuentra en la parte superior derecha de la página. Al pulsar, se abre un desplegable y se selecciona la opción de formato “*Tabular (text)*” para guardarse el archivo en formato TSV.

2.2. Ensembl

En las opciones de la parte superior de la página (*https://www.ensembl.org*), se selecciona “BioMart”, una herramienta web única de esta BBDD que permite la extracción de los datos de manera personalizada mediante la selección de atributos y filtros (Howe *et al.*, 2021).

Se selecciona, en primer lugar, el *data set* del cual se va a obtener la información: *“Ensembl Variation* 104” y *“Human Short Variants* (SNPs *and indels excluding flagged variants*) (GRCh38.p13)”.

Del total de filtros, únicamente se emplea uno, para seleccionar el fenotipo “Duchenne Muscular Dystrophy” o “Becker Muscular Dystrophy” En cuanto a los atributos, los cuales hacen referencia a las columnas que se quiere que tenga el fichero,

se seleccionan los siguientes: "*Variant name*", "*Variant source*", "*Chromosome/scaffold name*", "*Chromosome/scaffold position start (bp)*", "*Chromosome/scaffold position end (bp)*", "*Strand*", "*Variant alleles*", "*Associated variant risk allele*", "*Clinical significance*", "*Phenotype name*", "*Phenotype description*", "*P value*", "*Authors*" y "*Year*".

Una vez seleccionados los filtros y los atributos, se exporta en formato TSV.

2.3. LOVD

Esta BBDD no tiene opción de descargar las variaciones. Únicamente se pueden ver en la propia interfaz web de la página (*https://www.lovd.nl/*).

Tras tratar de ponerse en contacto con el responsable de la BBDD para solicitar un archivo con las variaciones, no se obtuvo respuesta. Por esta razón, en base a las recomendaciones de los expertos clínicos, se procedió a copiar las variaciones manualmente en una página de Excel.

El archivo Excel tiene formato XLSX, por lo que, para proceder a su tratamiento posterior, se exporta como archivo con formato TSV.

3. EXPLICACIÓN DE LOS *SCRIPTS*

3.1. Filtrado de los ficheros crudos

Lo primero que se realiza tras la extracción de los archivos, a partir de cada repositorio, es el filtrado de los ficheros crudos o sin tratar. Con este filtrado, se seleccionan las columnas de interés y se eliminan las redundancias que puedan haber. Los resultados de este filtrado se muestran en la **Tabla 1A**.

Tabla 1A. Ficheros de las tres BBDD, antes y después del proceso de filtrado.

BBDD	*Input* / Nº variaciones	Nombre *script*	*Output* / Nº variaciones
Ensembl	Duchenne_Ensembl.tsv / 1796	001_ensembl_D.py	001_Ensembl_filtered_D.tsv / 1425
	Becker_Ensembl.tsv / 137	001_ensembl_B.py	001_Ensembl_filtered_B.tsv / 71
ClinVar	Duchenne_ClinVar.tsv / 2894	002_clinvar_D.py	002_ClinVar_filtered_D.tsv / 2894
	Becker_ClinVar.tsv / 122	002_clinvar_B.py	002_ClinVar_filtered_B.tsv / 122
LOVD	Duchenne_Becker_LOVD.tsv / 30875	003_LOVD_D&B.py	003_LOVD_filtered_D&B.tsv / 7689

En la tabla anterior, se observan cuatro columnas. En la primera, se indica la BBDD de la que proceden los archivos; en la segunda, el nombre del archivo de entrada (o *input*) y el número de variaciones que incluye; en la tercera, el *script* por el que se hace pasar el *input*; y, en la cuarta, el fichero de salida *(output)*.

A continuación, se muestra el *script* “003_LOVD_D&B.py”, dado que se trata del que permite extraer información nueva que no se puede obtener haciendo uso de la plataforma OraGenDel. Los *scripts* “001_ensembl_D.py”, “001_ensembl_B.py”, “002_clinvar_D.py”, “002_clinvar_B.py” no se muestran en el presente manual, dado que realizan la misma función que el que se muestra a continuación pero para las BBDD de Ensembl y ClinVar.

```
# capturar el fichero tsv (Duchenne_Becker_LOVD.tsv)
import sys
tsvfile = sys.argv[1]

# definir la función main
def main(fichero):
    # seleccionar campos y crear diccionario con las significancias clini
cas
    diccionario, diccionario2, outfilee2 = funcion_seleccion(fichero)
    # seleccionar significancia asociada a la variacion
    outfilee3 = funcion_signific(diccionario, diccionario2, outfilee2)
    # eliminar redundancias en base al ID de la BBDD de LOVD
    outfilee4 = funcion_filtrado(outfilee3)

# crear fichero con los campos de interes
def funcion_seleccion(file):
    # crear string para quedarme con los nombres solo una vez
    unique = ''
    # crear diccionario para guardar ID de la BBDD y las distintas signif
icancias clinicas
    # key: identificador
    # value: significancias clinicas
    dic_sign = {}
    # crear diccionario para guardar toda la bibliografia asociada a cada
 variacion
    # key: identificador
    # value: referencias
    dic_biblio = {}
    # abrir fichero nuevo
    outfile2 = open('leiden_split.txt','w')
    # recorrer fichero linea a linea
    n = 0
    firstline = 'a'
    for line in open(file):
        line = line.strip('\n').split('\t')
        # primera linea
        if firstline:
            # escribir en el output las columnas: nombre_dbSNP - Leiden_I
D - Chr - Change - Location - Clinical significance - Effect - Exon - Ref
erence(s)
            outfile2.write('dbSNPname' + '\t' + 'DB-
ID' + '\t' + 'Chr' + '\t' + 'Change' + '\t' + 'Location_start' + '\t' + '
```

```
Location_end' + '\t' + 'Genome (hg38)' + '\t' + 'cDNA (NM_004006.2)' + '\
t' + 'Clinical significance' + '\t' + 'Effect' + '\t' + 'Exon' + '\t' + '
Reference(s)' + '\t' + 'Version' + '\t' + 'Proteina' + '\n')
            firstline = ''
            continue
        # definir conceptos
        identif = line[11]
        if '(recessive)' in line[6] and 'likely' in line[6]:
            signif = 'likely pathogenic'
        elif '(recessive)' in line[6] and 'likely' not in line[6]:
            signif = 'pathogenic'
        else:
            signif = line[6]
        fecha_ref = line[13]
        # añadir significancias clinicas y fecha a cada identificador
        if identif not in dic_sign.keys():
            dic_sign[identif] = []
            dic_biblio[identif] = []
        # escribir en el diccionario
        dic_sign[identif].append(signif)
        dic_biblio[identif].append(fecha_ref)

        # eliminar las repetidas
        if identif not in unique:
            n += 1
            unique = identif
            # de esa linea crear un fichero con nombre - chr (X todas por
que gen = DMD) - cambio - posicion - significancia clinica - referencia -
 version
            # definir campos
            dbSNPname = line[15]
            cDNA = line[2]
            effect = line[0]
            exon = line[1]
            location_change38 = line[8]
            location_change37 = line[7]
            change = ''
            location = ''
            proteina = line[4]


            # extraer la localizacion y el cambio version GRCh38
            if 'g.?' not in location_change38 or '-
' not in location_change38:

                # modificar el campo de posicion (GRCh38)
                for i in range(len(location_change38)):
                    # si es un "indel"
                    u = 0
```

```
                    if 'delins' in location_change38 and '"' not in locat
ion_change38:
                        u += 1
                        if u == 1:
                            # cuando aparezca la "d" de "delins"
                            d = 0
                            if 'd' in location_change38:
                                d += 1
                                if d == 1:
                                    if location_change38[i] == 'd': #or l
ocation_change[i+1] != '[':
                                        change = location_change38[i:]
                                        location = location_change38[2:i]
                    # si hay comillas significa que hay anotados más de u
n tipo de cambio
                    elif 'delins' in location_change38 and '"' in locatio
n_change38:
                        u += 1
                        if u == 1:
                            change = 'delins + (...)'
                    # pero pese a que haya comillas y más de un tipo de c
ambio, este no tiene porqué ser un "indel"
                    elif ']"' in location_change38:
                        u += 1
                        if u == 1:
                            change = 'varios cambios (...)'
                    # si es una insercion
                    elif 'delins' not in location_change38:
                        m = 0
                        if 'ins' in location_change38:
                            m += 1
                            if m == 1:
                                if location_change38[i] == 'i':
                                    change = location_change38[i:]
                                    location = location_change38[2:i]
                        else:
                            change = location_change38[-3:]
                            location = location_change38[2:-3]
                    # si no hay informacion, poner un guion
                    elif '-' in location_change38:
                        change = '-'
                        location = '-'
                    else:
                        change = location_change38[-3:]
                        location = location_change38[2:-3]
                #print(location_change + '   &   ' + change + '   &   ' +
 location) # it works

                # editar campo "location"
```

```
                location_ini = ''
                location_end = ''
                b = 0
                for i in range(len(location)):
                    # si no hay parentesis ni barra baja
                    if '(' not in location and '_' not in location:
                        location_ini = location
                        location_end = location
                    # si no hay parentesis pero si hay barra baja
                    elif '(' not in location and '_' in location:
                        if location[i] == '_':
                            location_ini = location[:i]
                            location_end = location[i+1:]
                    # si hay parentesis
                    if '(' in location and '_' in location:
                        if location[i] == '_':
                            b += 1
                            if b == 1:
                                if location[1:i] == '?':
                                    location_ini = location[i+1:i+9]
                                else:
                                    location_ini = location[1:i]
                            elif b == 3:
                                if location[i+1] == '?':
                                    location_end = location[i-8:i]
                                else:
                                    location_end = location[i+1:-1]
                #print(location + '   &   ' + location_ini + '   &   ' +
location_end) # it works
                # escribir en el output
                outfile2.write(dbSNPname + '\t' + identif + '\t' + 'X' +
'\t' + change + '\t' + location_ini + '\t' + location_end + '\t' + locati
on_change38 + '\t' + cDNA + '\t' + signif + '\t' + effect + '\t' + exon +
 '\t' + fecha_ref + '\tGRCh38\t' + proteina + '\n')

            ######## extraer la localizacion y el cambio version GRCh37
            if ('g.?' in location_change38 or '-
' in location_change38) and ('g.?' not in location_change37 or '-
' not in location_change37):
                # modificar el campo de posicion (GRCh37)
                for i in range(len(location_change37)):
                    # si es un "indel"
                    u = 0
                    if 'delins' in location_change37 and '"' not in locat
ion_change37:
                        u += 1
                        if u == 1:
                            # cuando aparezca la "d" de "delins"
                            d = 0
```

```
                        if 'd' in location_change37:
                            d += 1
                            if d == 1:
                                if location_change37[i] == 'd': #or l
ocation_change[i+1] != '[':
                                    change = location_change37[i:]
                                    location = location_change37[2:i]
                                #else:
                                #    change = 'delins'
                                #    location = location_change[2:]
                    # si hay comillas significa que hay anotados más de u
n tipo de cambio
                    elif 'delins' in location_change37 and '"' in locatio
n_change37:
                        u += 1
                        if u == 1:
                            change = 'delins + (...)'
                    # pero pese a que haya comillas y más de un tipo de c
ambio, este no tiene porqué ser un "indel"
                    elif ']"' in location_change37:
                        u += 1
                        if u == 1:
                            change = 'varios cambios (...)'
                    # si es una insercion
                    elif 'delins' not in location_change37:
                        m = 0
                        if 'ins' in location_change37:
                            m += 1
                            if m == 1:
                                if location_change37[i] == 'i':
                                    change = location_change37[i:]
                                    location = location_change37[2:i]
                        else:
                            change = location_change37[-3:]
                            location = location_change37[2:-3]
                    # si no hay informacion, poner un guion
                    elif '-' in location_change37:
                        change = '-'
                        location = '-'
                    else:
                        change = location_change37[-3:]
                        location = location_change37[2:-3]
                #print(location_change + '   &   ' + change + '   &   ' +
 location) # it works

                # editar campo "location"
                location_ini = ''
                location_end = ''
                b = 0
```

```
                for i in range(len(location)):
                    # si no hay parentesis ni barra baja
                    if '(' not in location and '_' not in location:
                        location_ini = location
                        location_end = location
                    # si no hay parentesis pero si hay barra baja
                    elif '(' not in location and '_' in location:
                        if location[i] == '_':
                            location_ini = location[:i]
                            location_end = location[i+1:]
                    # si hay parentesis
                    if '(' in location and '_' in location:
                        if location[i] == '_':
                            b += 1
                            if b == 1:
                                if location[1:i] == '?':
                                    location_ini = location[i+1:i+9]
                                else:
                                    location_ini = location[1:i]
                            elif b == 3:
                                if location[i+1] == '?':
                                    location_end = location[i-8:i]
                                else:
                                    location_end = location[i+1:-1]
                #print(location + '   &   ' + location_ini + '   &   ' +
location_end) # it works
                # escribir en el output
                outfile2.write(dbSNPname + '\t' + identif + '\t' + 'X' +
'\t' + change + '\t' + location_ini + '\t' + location_end + '\t' + locati
on_change37 + '\t' + cDNA + '\t' + signif + '\t' + effect + '\t' + exon +
 '\t' + fecha_ref + '\tGRCh37\t' + proteina + '\n')

            if ('g.?' in location_change38 or '-
' in location_change38) and ('g.?' in location_change37 or '-
' in location_change37):
                change = '-'
                location_ini = '-'
                location_end = '-'
                outfile2.write(dbSNPname + '\t' + identif + '\t' + 'X' +
'\t' + change + '\t' + location_ini + '\t' + location_end + '\t' + locati
on_change37 + '\t' + cDNA + '\t' + signif + '\t' + effect + '\t' + exon +
 '\t' + fecha_ref + '\tNA\t' + proteina + '\n')
    outfile2.close()

    print(str(n) + ' variaciones unicas')

    return(dic_sign,dic_biblio,outfile2)
```

```
def funcion_signific(dicc,dicc2, output2):
    #print(dic_sign)
    mas = 0
    # diccionario para guardar 1 vez las significancias clinicas distinta
s que hay asociadas a cada variacion
    # key: identificador de la BBDD
    # value: significancias
    dic_uniq = {}
    # diccionario para guardar toda la bibliografia asociada a 1 variacio
n
    # key: identificador de la BBDD
    # value: refs
    dic_bib = {}
    # diccionario para contar cuantas veces ha sido cargada cada variacio
n en la BBDD
    # key: identificador de la BBDD
    # value: nº de veces que aparece
    dic_num = {}
    for key in dicc:
        # contar cuantas veces ha sido cargada en la BBDD la misma variac
ion
        u = 0
        uniq = []
        dic_uniq[key] = ''
        dic_num[key] = ''
        for signifi in dicc[key]:
            u += 1
            #dic_num[signifi] == ''
            # si no se repite la significancia
            if signifi not in uniq:
                uniq.append(signifi)
                # añadir al diccionario cada tipo de significancia solo 1
 vez
                if dic_uniq[key] == '':
                    #dic_num[signifi] == str(m)
                    dic_uniq[key] += signifi
                elif dic_uniq[key] != '':
                    #dic_num[signifi] == str(m)
                    dic_uniq[key] += ', ' + signifi
        dic_num[key] = str(u)
        #print(str(u) + ' elementos, ' + str(len(uniq)) + ' significancia
(s) distinta(s), que son: ')
        if u != 0:
            mas += 1
        #print(key, str(u), dic_sign[key])
    #print(str(mas) + ' variaciones con >1 significancia clinica')
    #print(dic_uniq)
    #print(dic_num)
```

```
    for key in dicc2:
        dic_bib[key] = ''
        unique = []
        for element in dicc2[key]:
            # si no se repite la referencia bibliografica
            if element not in unique:
                unique.append(element)
                # añadir al diccionario cada tipo de significancia solo 1
 vez
                if dic_bib[key] == '':
                    dic_bib[key] += element
                elif dic_bib[key] != '':
                    dic_bib[key] += ';; ' + element
    #print(dic_bib)

    # generar un nuevo output
    outfile3 = open('leiden_split_2.txt','w')
    # abrir el output2
    firstline = 'a'
    # diccionario
    # key: id
    # value: 1er campo localizacion/ version cromosoma?
    dic_version = {}
    for line in open('leiden_split.txt'):
        line = line.strip('\n').split('\t')
        if firstline:
            firstline = ''
            outfile3.write(line[0] + '\t' + line[1] + ' (veces)\t' + line
[2] + '\t' + line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] +
 '\t' + line[7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t
' + line[11] + '\t' + line[12] + '\t' + line[13] + '\n')
            continue
        # modificar el campo de significancias clinicas, para incluir tod
as las asociadas a una misma variacion
        if '37' in line[12] and len(line[4]) > 5:
            outfile3.write(line[0] + '\t' + line[1] + ' (' + dic_num[line
[1]] + ')\t' + line[2] + '\t' + line[3] + '\t' + line[4] + '\t' + line[5]
 + '\t' + line[6] + '\t' + line[7] + '\t' + dic_uniq[line[1]] + '\t' + li
ne[9] + '\t' + line[10] + '\t' + dic_bib[line[1]] + '\t' + line[12] + '\t
' + line[13] + '\n')
        #elif '38' in line[12] and len(line[4]) < 4:
        #    continue
        elif '38' in line[12] and len(line[4]) > 5:
            outfile3.write(line[0] + '\t' + line[1] + ' (' + dic_num[line
[1]] + ')\t' + line[2] + '\t' + line[3] + '\t' + line[4] + '\t' + line[5]
 + '\t' + line[6] + '\t' + line[7] + '\t' + dic_uniq[line[1]] + '\t' + li
ne[9] + '\t' + line[10] + '\t' + dic_bib[line[1]] + '\t' + line[12] + '\t
' + line[13] + '\n')
        else:
```

```
            print(line[1])

    outfile3.close()
    return(outfile3)

def funcion_filtrado(output3):
    # generar un nuevo output
    outfile4 = open('003_LOVD_filtered_D&B.tsv','w')
    # abrir el output2
    firstline = 'a'
    unico = []
    for line in open('leiden_split_2.txt'):
        line = line.strip('\n').split('\t')
        if firstline:
            firstline = ''
            outfile4.write(line[0] + '\t' + line[1] + '\t' + line[2] + '\
t' + line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] + '\t' +
line[7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t' + line
[11] + '\t' + line[12] + '\t' + line[13] + '\n')
            continue
        if line[1] not in unico:
            unico.append(line[1])
            outfile4.write(line[0] + '\t' + line[1] + '\t' + line[2] + '\
t' + line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] + '\t' +
line[7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t' + line
[11] + '\t' + line[12] + '\t' + line[13] + '\n')
    outfile4.close()
    return(outfile4)

main(tsvfile)
```

3.2. Uniformización del nombre

Una vez generados los ficheros que contienen las columnas de interés, y a los cuales se les han eliminado las posibles redundancias que pudiesen haber, se crea una nueva columna para nombrar las variaciones con el nombre que sigue la estructura propuesta por el Oráculo Genómico de Delfos:

'Localización : Cambio (Ensamblaje)'

Los *scripts* desarrollados para ello se denominan:

- 011_OraGenDel name_ensembl.py
- 012_OraGenDel name_clinvar.py
- 013_OraGenDel name_LOVD.py

A estos últimos, se les hizo pasar el *output* generado en el paso anterior, pero esta vez como *input*. El resultado consistió en la obtención de los ficheros con la misma información que antes de pasar por los *scripts*, pero con una nueva columna añadida en la cual se mostraba el nombre de cada variación en formato OraGenDel.

A continuación, se muestra el *script* "013_OraGenDel name_LOVD.py" que se desarrolló con tal fin.

```
# capturar fichero input (003_LOVD_filtered_D&B.tsv)
import sys
tsvfile = sys.argv[1]

def main(fichero):
    # escribir nombre de las variaciones como en OraGenDe (Chr:position:a
lleles)
    delfos_name = funcion_nombre(fichero)

def funcion_nombre(file):
    firstline = 'a'
    # abrir output
    out = open('013_LOVD_name_D&B.tsv','w')
    d = 0
    g = 0
    f = 0
    dif = ''
    for line in open(file):
        line = line.strip('\n').split('\t')
        cambio = ''
        ref = ''
        alt = ''
        oragende = ''
        # escribir linea header
        if firstline:
            out.write('OraGenDe\t' + line[0].replace('dbSNPname','dbSNP_I
D') + '\t' + line[1] + '\t' + line[2] + '\t' + line[3] + '\t' + line[4] +
 '\t' + line[5] + '\t' + line[6] + '\t' + line[7] + '\t' + line[8] + '\t'
```

```
 + line[9] + '\t' + line[10] + '\t' + line[11] + '\t' + line[12] + '\t' +
 line[13] + '\n')
            firstline = ''
            continue
        # campo localización
        if len(line[4]) > 3 and len(line[5]) > 3 and 'AC[15_25]' not in l
ine[5]:
            inicio = line[4]
            final = line[5]
            # calcular diferencia entre localizacion final e inicial
            #print(inicio, final)
            dif = int(final) - int(inicio)
            #print(dif)
            # si la diferencia es > 10 ntds
            if dif > 10 or dif < -10:
                cambio = 'longchange'
                oragende = 'ChrX:g.' + line[4] + '-
' + line[5] + ':' + cambio + '(' + line[12] + ')'
                f += 1
            else:
                cambio = line[3]
                # si hay dos alelos
                if '>' in line[3]:
                    g += 1
                    oragende = 'ChrX:g.' + line[4] + '-
' + line[5] + ':' + cambio + '(' + line[12] + ')'
                elif '>' not in line[3] and '?' not in line[3] and '[' no
t in line[3]:
                    #print(line[3])
                    d += 1
                    oragende = 'ChrX:g.' + line[4] + '-
' + line[5] + ':?>?(' + line[12] + ')'
                else:
                    #f += 1
                    print(line[1], line[3])
                    oragende = '_ChrX:g.' + line[4] + '-
' + line[5] + ':' + cambio + '(' + line[12] + ')'

        out.write(oragende + '\t' + line[0] + '\t' + line[1] + '\t' + lin
e[2] + '\t' + line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6]
+ '\t' + line[7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\
t' + line[11] + '\t' + line[12] + '\t' + line[13] + '\n')

    print(g)
    print(d)
    print(f)
    out.close()
    return(out)
main(tsvfile)
```

3.3. Clasificación de las variaciones

La clasificación de las variaciones se llevó a cabo tanto para las variaciones extraídas de Ensembl y ClinVar como para las de LOVD. Dicha clasificación se basó en seleccionar aquellas cuya significancia clínica era "*pathogenic*" o "*likely pathogenic*", debido a que este era uno de los requisitos necesarios para ser aceptadas por el algoritmo de clasificación de la plataforma OraGenDel (lo cual se explica en el **Subapartado 4.3.1**), entre otros requisitos. Cabe destacar que las variaciones que albergaban la significancia "*VUS*" y "*pathogenic*", o "NA" y "*pathogenic*" también fueron incluidas dentro de la clasificación, para no descartar ninguna posibilidad.

Los *scripts* desarrollados para extraer las variaciones relevantes de ClinVar y Ensembl consiguieron obtener un 96.72% de cobertura al compararlas con las extraídas por OraGenDel para BMD y un 99.75% para las de DMD, lo que permitió asegurar la fiabilidad de los *scripts* a la hora de extraer las variaciones para la BBDD de LOVD.

A continuación, se muestra el *script* que se elaboró para realizar dicha clasificación para la BBDD de LOVD, seguido del resultado de la misma, el cual se puede observar en la **Figura 2A**.

```
# capturar fichero input (013_LOVD_name_D&B_.tsv)
import sys
file = sys.argv[1]

# contadores para la clasificacion
n = 0
p = 0
b = 0
g = 0
v = 0
a = 0
u = 0
s = 0
o = 0

# abrir fichero output
out = open('113_LOVD_D&B.tsv','w')
firstline = 'a'
for line in open(file):
    line = line.strip('\n')
    # si es la primera linea
    if firstline:
        firstline = ''
        # escribir la línea header del output
        out.write(line + '\n')
        continue
    # definir conceptos
    line = line.split('\t')
    sign = line[9]
    name = line[0]
```

```
    # escribir en el output aquellas variaciones que sean patogénicas o
probablemente patogenicas
    if ',' in sign:
        o += 1
        if 'pathogenic, likely pathogenic' in sign or 'likely pathogenic,
 pathogenic' in sign:
            s += 1
            out.write(line[0] + '\t' + line[1] + '\t' + line[2] + '\t' +
line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] + '\t' + line[
7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t' + line[11]
+ '\t' + line[12] + '\t' + line[13] + '\t' + line[14] + '\n')
        if 'pathogenic' in sign and 'benign' not in sign:
            s += 1

    if ',' not in sign:
        if 'pathogenic' in sign and 'likely pathogenic' not in sign:
            n += 1
            out.write(line[0] + '\t' + line[1] + '\t' + line[2] + '\t' +
line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] + '\t' + line[
7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t' + line[11]
+ '\t' + line[12] + '\t' + line[13] + '\t' + line[14] + '\n')
        if 'likely pathogenic' in sign:
            p += 1
            out.write(line[0] + '\t' + line[1] + '\t' + line[2] + '\t' +
line[3] + '\t' + line[4] + '\t' + line[5] + '\t' + line[6] + '\t' + line[
7] + '\t' + line[8] + '\t' + line[9] + '\t' + line[10] + '\t' + line[11]
+ '\t' + line[12] + '\t' + line[13] + '\t' + line[14] + '\n')
        if 'benign' in sign and 'likely benign' not in sign:
            b += 1
        if 'likely benign' in sign:
            g += 1
        if 'VUS' in sign:
            v += 1
        if 'NA' in sign:
            a += 1
        if 'unclassified' in sign:
            u += 1
# sacar por pantalla el numero total de cada tipo de variaciones atendiendo
a su clasificación
print(str(o) + ' variaciones con más de una interpretación')
print(str(n) + ' pathogenic')
print(str(p) + ' likely pathogenic')
print(str(b) + ' benign')
print(str(g) + ' likely benign')
print(str(v) + ' VUS')
print(str(a) + ' NA')
print(str(u) + ' unclassified')
print(str(s) + ' con >1 significancia pero patogénicas principalmente')
```

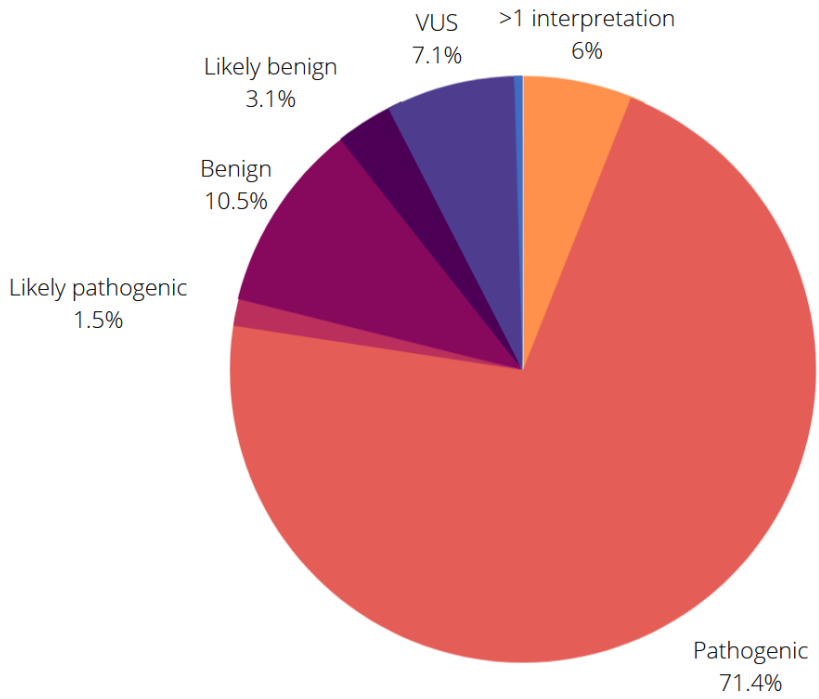

Figura 2A. Clasificación de las variaciones de LOVD según su significancia clínica.

El resultado de la clasificación que se muestra en la Figura 2A consistió en 4970 variaciones "Pathogenic" (71.4%), 103 variaciones "Likely pahogenic" (1.5%), 714 variaciones "Benign" (10.5%), 484 variaciones "VUS" (7.1%), 410 variaciones con más de una interpretación (6%), 210 variaciones "Likely benign" (3.1%) y 32 variaciones "NA"/"Unclassified" (<1%).

3.4. Otros análisis

Por último, una vez analizadas las características y el tipo de variaciones de la BBDD de LOVD, y habiendo determinado su importancia en base al bajo ratio de cobertura que presentaba esta última frente a las BBDD de Ensembl y ClinVar, se plantearon una serie de estudios con tal de determinar el significado biológico que podía extraerse del estudio de las variaciones.

Para llevar a cabo dichas determinaciones, se estudiaron las variaciones atendiendo al tipo de cambio y a la localización. El siguiente *script* analiza los tipos de cambios de las variaciones clasificadas como relevantes para la DMD y para la BMD. Los resultados de este estudio se muestran en el **Subapartado 5.4.1**.

```
# capturar archivo input (variaciones relevantes para DMD/BMD)
import sys
file = sys.argv[1]
# contador del tipo de cambio asociado a cada variacion
totalsnv = 0
total = 0
snv = 0
ter = 0
miss = 0
```

```
fs = 0
dup = 0
dele = 0
microsat = 0
indel = 0
loss = 0
ins = 0
inv = 0
interr = 0
cor = 0

firstline = 'a'
for line in open(file):
    line = line.strip('\n').split('\t')
    # definir las variables
    cambio = line[1]
    proteina = line[17]
    if firstline:
        firstline = ''
        continue
    total += 1
    if 'Indel' in cambio:
        indel += 1
    if 'single nucleotide variant' in cambio:
        snv += 1
        totalsnv += 1
        # analizar si se trata de una alteracion de tipo nonsense o
missense
        if 'Ter' in proteina:
            ter += 1
        if '=' in proteina:
            miss += 1
        if '=' not in proteina and 'fs' not in proteina and 'Ter' not in
proteina:
            miss += 1
    if 'Deletion' in cambio:
        dele += 1
    if 'Duplication' in cambio:
        dup += 1
    if 'Microsatellite' in cambio:
        microsat += 1
    if 'loss' in cambio:
        loss += 1
    if 'Insertion' in cambio:
        ins += 1
    if 'inv' in cambio:
        inv += 1
    if cambio == '?':
        interr += 1
```

```
    if '[' in cambio or ']' in cambio:
        cor += 1
    # ver cuantas variaciones del total afectan a la pauta de lectura
    if 'fs' in proteina:
        fs += 1

print(totalsnv)

# sacar por pantalla los resultados de la clasificación según el tipo de
cambio, también su porcentaje con respecto al total de variaciones
print(str(total) + ' variaciones en total (' + str(round(total/total*100,
2)) + ')')
print(str(snv) + ' SNPs (' + str(round(snv/total*100,2)) + '), que se cla
sifican en:')
print(str(round(totalsnv/totalsnv*100,2)))
print(str(ter) + ' nonsense (' + str(round(ter/totalsnv*100,2)) + ')')
print(str(miss) + ' missense (' + str(round(miss/totalsnv*100,2)) + ')')

print(str(dup) + ' duplications (' + str(round(dup/total*100,2)) + ')')
print(str(dele) + ' deletions (' + str(round(dele/total*100,2)) + ')')
print(str(microsat) + ' microsatellites (' + str(round(microsat/total*100
,2)) + ')')
print(str(indel) + ' indels (' + str(round(indel/total*100,2)) + ')')
print(str(loss) + ' copy number loss (' + str(round(loss/total*100,2)) +
')')
print(str(ins) + ' insertions (' + str(round(ins/total*100,2)) + ')')
print(str(inv) + ' inversions (' + str(round(inv/total*100,2)) + ')')
print(str(interr) + ' ? (' + str(round(interr/total*100,2)) + ')')
print(str(cor) + ' corchete (' + str(round(cor/total*100,2)) + ')\n')
print(str(fs) + ' variaciones que desplazan el marco de lectura o framesh
ift variations (' + str(round(fs/total*100,2)) + ')')
```

El siguiente *script*, por último, se centra en determinar la localización de las variaciones a lo largo de la secuencia genómica del gen. Los resultados de dicho análisis se muestran en las gráficas del **Subapartado 5.4.2**.

```
# capturar fichero input
import sys
exonslocation = sys.argv[1] # archivo con las localizaciones de los
exones
variationsfile = sys.argv[2] # fichero con variaciones de DMD/BMD

def main(exones,fichero):
    # crear diccionario con los exones y su localización
    diccionario = funcion_dicc(exones)
    # asociar cada variacion con su localizacion en el genoma
    dic1,dic2 = funcion_ubicar(diccionario,fichero)
```

```
    # contar cuantas variaciones hay asociadas a cada localización
    dic_contador = funcion_contar(dic1,dic2)

def funcion_dicc(exonsfile):
    # diccionario
    # key: exon
    # value: localizacion inicial y final
    dic_exons = {}
    # abrir el fichero con los exones y sus posiciones genómicas
    firstline = 'a'
    for line in open(exonsfile):
        line = line.strip('\n').split('\t')
        if firstline:
            firstline = ''
            continue
        nexon = line[0]
        ini = line[3]
        fin = line[4]
        dic_exons[nexon] = ini, fin
    #print(dic_exons)
    return(dic_exons)

def funcion_ubicar(diccio,variaciones):
    # crear diccionario con los exones y las variaciones que se asocian a
 cada uno
    # key: exones
    # value: variaciones
    dic_exones = {}
    # asignar las llaves del diccionario
    for i in range(79):
        dic_exones[i+1] = []
    # crear diccionario con los intrones y las variaciones que se asocian
 a cada uno
    # key: intrones
    # value: variaciones
    dic_intrones = {}
    # asignar las llaves del diccionario
    for j in range(78):
        dic_intrones[j+1] = []

    firstline = 'a'
    # abrir fichero con las variaciones
    for line in open(variaciones):
        line = line.strip('\n').split('\t')
        if firstline:
            firstline = ''
            continue
        inicial = line[0][7:15]
        final = line[0][16:24]
```

```
        #print(final)
        # abrir diccionario
        stop = 'a'
        stop2 = 'a'
        n = 0
        for exon in diccio:
            #print(exon)
            n += 1
            posiciones = diccio[str(n)]
            # guardar
            pos = ''
            # recorrer las 2 posiciones de cada exon
            t = 0
            p = 0
            for posicion in posiciones:
                p += 1
                # si es la posicion inicial
                if p == 1:
                    # si la localización inicial de la variacion es mayor
 que la posición donde empieza el exón
                    if inicial > posicion:
                        pos = inicial
                        #print(exon,loc)
                        continue
                    if inicial < posicion and stop:
                        #print(dic_intrones)
                        t = 1
                        stop = ''
                        dic_intrones[n].append(line[0])
                if p == 2:
                    # si la localización inicial de la variacion es menor
 que la posicion final
                    if pos > posicion:
                        continue
                    if pos < posicion and stop2:
                        stop2 = ''
                        if line[0] not in dic_intrones.values():
                            dic_exones[n].append(line[0])
    print(dic_intrones)
    #print(dic_exones)
    return(dic_intrones,dic_exones)

def funcion_contar(intrones,exones):
    # crear diccionario para contar el numero de variaciones asociado a c
ada intron
    # key: intron
    # value: numero de variaciones asociadas
    dic_counti = {}
```

```
    # crear diccionario para contar el numero de variaciones asociado a c
ada intron
    # key: intron
    # value: numero de variaciones asociadas
    dic_counte = {}
    # asociar variaciones con intrones
    intr = 0
    for key in introns:
        dic_counti[key] = ''
        n = 0
        for variacion in introns[key]:
            intr += 1
            n += 1
            dic_counti[key] = n
    print(dic_counti)
    print(intr)
    # asociar variaciones con exones
    exo = 0
    for key in exones:
        dic_counte[key] = ''
        m = 0
        for variacion in exones[key]:
            exo += 1
            m += 1
            dic_counte[key] = m
    print(dic_counte)
    print(exo)

main(exonslocation,variationsfile)
```

ANEXO 2

- MCGH v.3 -

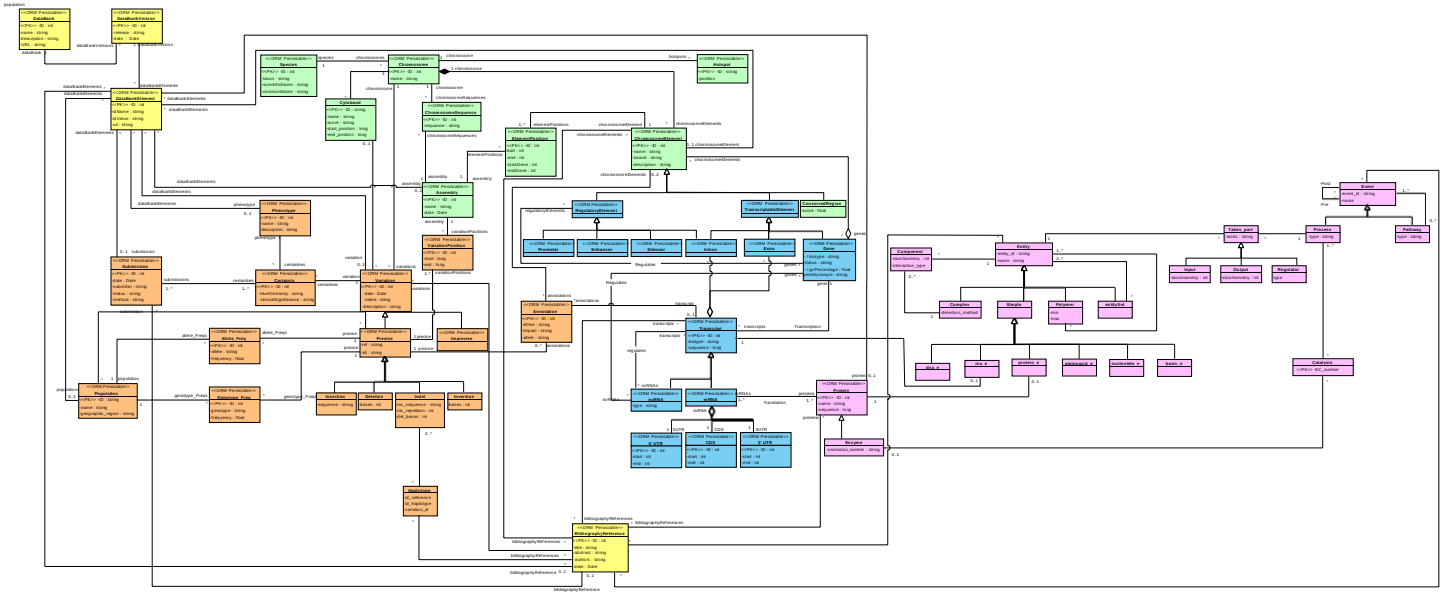

Figura 1B. Modelo Conceptual del Genoma Humano. Fuente: Reyes, 2018.