Anotación de nuevos microRNAs en el genoma porcino mediante una aproximación basada en Machine Learning

Emilio Mármol Sánchez
Máster universitario en Bioinformática y Bioestadística (UOC-UB).
Área del trabajo final: *Machine Learning*

Albert Pla Planas
Jose Antonio Morán Moreno

Junio 2018

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Anotación de nuevos microRNAs en el genoma porcino mediante una aproximación basada en Machine Learning*
Nombre del autor:	*Emilio Mármol Sánchez*
Nombre del consultor/a:	*Albert Pla Planas*
Nombre del PRA:	*Jose Antonio Morán Moreno*
Fecha de entrega (mm/aaaa):	06/2018
Titulación:	*Máster en Bioinformática y Bioestadística*
Área del Trabajo Final:	*Machine Learning*
Idioma del trabajo:	*Castellano*
Palabras clave	*Machine Learning, miRNA, Support Vector Machine*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

La predicción computacional de microRNAs (miRNAs) supone un campo de investigación activo en la actualidad, sobre todo en especies no modelo cuyas anotaciones son aún limitadas y poco fiables. Mediante la utilización de una aproximación basada en algoritmos de Machine Learning como el *Support Vector Machine* (SVM) y *Random Forest* (RF), y haciendo uso de la comparación por homología en la anotación de miRNAs en humano, hemos desarrollado un proceso para la identificación y anotación de nuevos candidatos a estructuras pre-miRNA en el genoma porcino. Partiendo de la generación de un set de datos positivos y negativos, filtrados según tamaño y conformación estructural, se definieron diversos atributos estructurales para cada secuencia, con el objetivo de entrenar un modelo SVM de *Machine Learning*. Un conjunto de secuencias candidatas obtenidas mediante comparación por homología, fueron clasificadas como candidatos pre-miRNAs por el modelo SVM entrenado previamente, y posteriormente filtradas mediante un análisis de fiabilidad de posición genómica (*Neighbouring Score*). Mediante este proceso fuimos capaces de identificar un total de 26 nuevas secuencias pre-miRNA candidatas en el genoma porcino. De entre ellas destacó el miRNA ssc-miR-483, homólogo del miRNA homónimo en humano hsa-miR-483, alojado en el intrón 2 del gen IGF2, cuya función estaría ligada a la regulación de la proliferación celular y la diferenciación de adipocitos, influyendo en la capacidad de integración y depósito de lípidos en respuesta a variaciones en la ingesta de alimentos. Estos resultados podrían ampliar el conocimiento sobre la regulación del metabolismo energético y lipídico en la especie porcina.

Abstract (in English, 250 words or less):

Computational discovery of microRNAs (miRNAs) poses a big research challenge nowadays, especially considering non-model species that lack accurate and reliable miRNA annotation. Through the application of a Machine Learning approach by using algorithms like Support Vector Machine (SVM) and Random Forest (RF) and making use of a homology-based comparison with miRNA annotation un humans, we developed a pipeline for identifying and annotating new pre-miRNA candidates in the porcine genome. We generated a set of positive and negative data, filtered considering size and structural folding, and then calculated a series of structural features for each considered sequence that where subsequently used for training a Machine Learning-based SVM classifier.

We extracted a set of candidate sequences in the porcine genome that showed to be homologous from human miRNA annotation and classified them by using the previously trained SVM model. These candidate pre-miRNAs sequences were then filtered according to a neighbouring feasibility analysis.

Our approach allowed us to identify 26 putative non-annotated pre-miRNA sequences in the porcine genome. Among them, we highlighted the putative candidate ssc-miR-483, homologous of human hsa-miR-483 and located at intron 2 of IGF2 gene. This miRNA has been associated to the regulation of cellular proliferation and adipocyte differentiation, modulating lipid integration and storage in response to food intake. These results could enhance our understanding of energy and lipid metabolism regulation in the porcine species.

Índice

Lista de figuras

Lista de Tablas

1. Introducción

1.1. Contexto teórico

MicroRNAs

Los microRNAs (miRNAs)[1] conforman un tipo particular de ARN no codificante que, en su forma biológicamente funcional, están formados por cadenas de ARN monocatenario de tamaño pequeño (18 a 22 nucleótidos). Se estima que los miRNAs suponen aproximadamente el 1% de los genes descritos en la actualidad[2], y actúan principalmente como reguladores de la expresión de ARNs mensajeros (mRNAs)[3]. Por otra parte, se sabe que determinados microRNAs pueden actuar sobre decenas de transcritos al mismo tiempo, interviniendo en la regulación de multitud de procesos biológicos y estando relacionados por lo tanto, con numerosos procesos y patologías, entre ellas, el desarrollo, regulación y progresión de procesos tumorales[4,5].

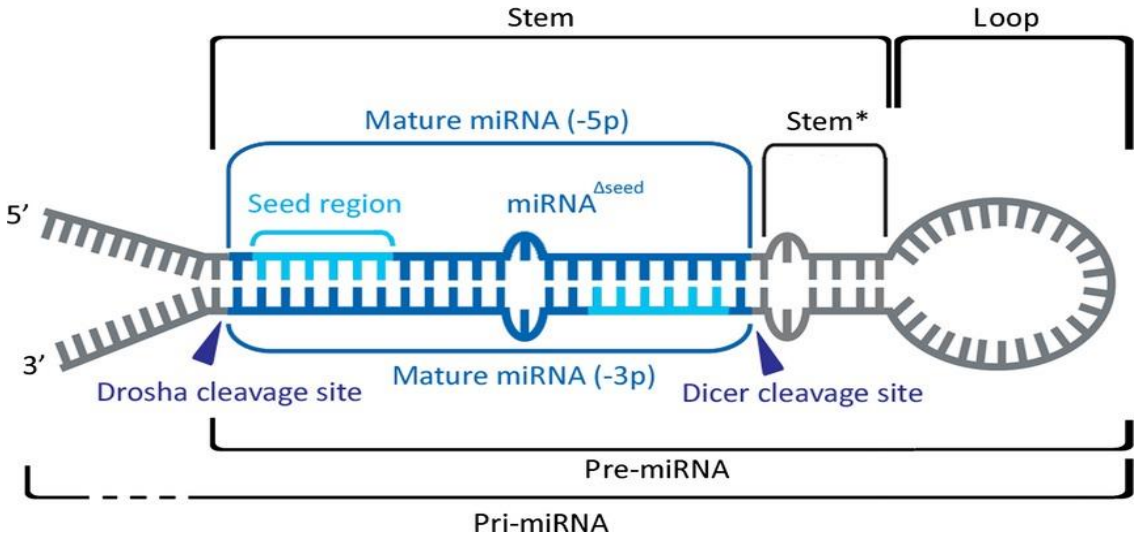

Figura 1: Estructura típica de un precursor de microRNA. Dos cadenas o *stems*, unidas por un *terminal loop*, que engloban las dos cadenas 5' y 3' que constituyen los microRNAs maduros. La región *seed* se corresponde con la zona metabólicamente funcional del microRNA. Jevsinek *et al.* 2013[6].

A pesar del extenso conocimiento que se ha ido acumulando sobre la función biológica[7], biogénesis[8] y posibilidades terapéuticas[9] que pueden aportar este tipo de ARN no codificante, la mayoría de estudios se han centrado en ensayos en especies modelo, particularmente en humano y ratón, siendo aún muy incipiente el conocimiento que existe sobre la diversidad biológica de los microRNAs en otras especies. Este hecho se pone de manifiesto en la extensa bibliografía y disponibilidad de datos sobre la función de los miRNAs, sus interacciones y particularidades, especialmente centrados en especies modelo, pero siendo ésta, tremendamente escasa para otras

especies menos estudiadas o para las cuales la atención científica no ha sido aún lo suficientemente amplia. Un caso paradigmático de este caso puede ser la especie porcina, en la cual existe aún un escaso y limitado conocimiento sobre el papel que los microRNAs cumplen en su biología y metabolismo, así como en las posibilidades que éstos pudieran brindar en la mejora genética de esta especie, de gran interés económico y ganadero.

Los miRNAs pueden encontrarse no sólo en el interior celular, donde mayoritariamente se alojan y cumplen su función biológica, sino que también han podido ser detectados en entornos extracelulares como medios de cultivo celular y diversos fluidos biológicos como sangre o esperma, conocidos en este caso como miRNAs circulantes[10]. Cabe destacar, por otra parte, las particularidades observadas en cuanto al procesado y maduración de los microRNAs en diferentes reinos biológicos. Por ejemplo, en el caso de las plantas, los miRNAs son madurados por completo en el entorno nuclear, y transportados al citoplasma posteriormente, donde fundamentalmente ejercen su función biológica mediante el *cleavage* de mRNAs diana, debido a su unión por complementariedad casi perfecta o completamente perfecta. Este hecho difiere fundamentalmente en el caso de los animales, donde los miRNAs presentan una complementariedad mucho más laxa respecto a sus mRNAs diana, interfiriendo en su translación o facilitando su degradación[11].

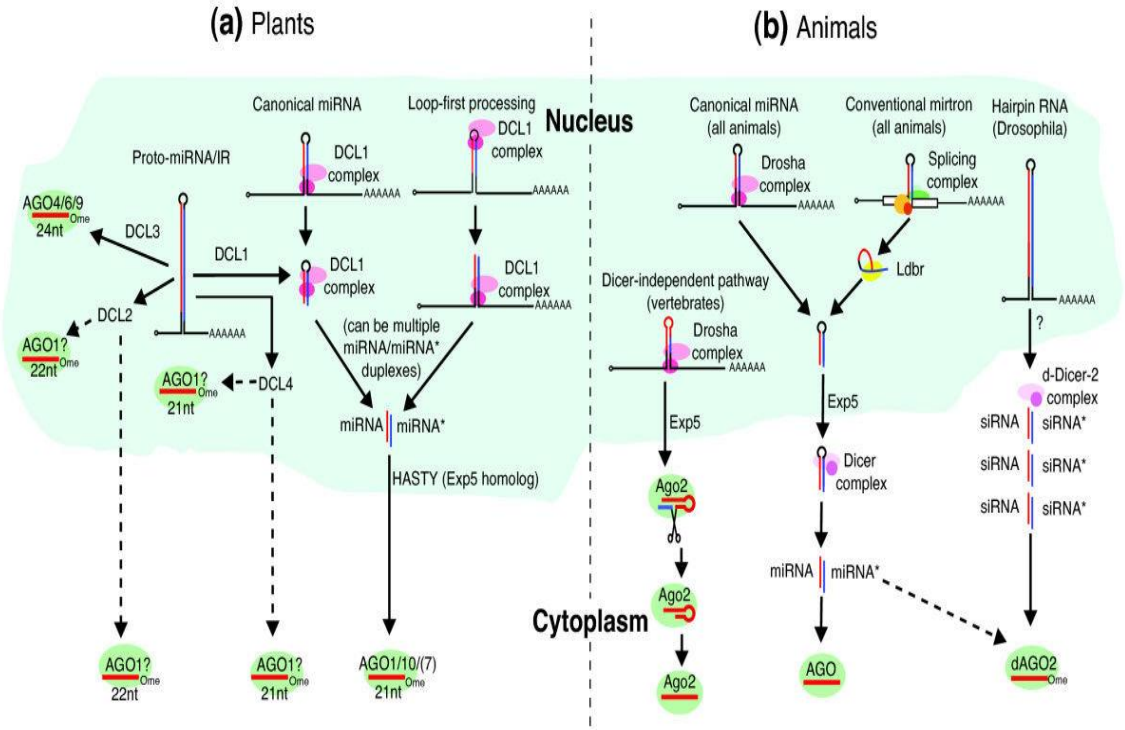

Figura 2: Diferencias entre el proceso de biogénesis de microRNAs en plantas y animales. **A)** En plantas, los microRNAs son directamente producidos por DCL1 (RNAse III Dicer-like1) y posteriormente transportados al citoplasma por HASTY (homólogo de Exp5), para finalmente ser introducidas en complejos proteicos junto con diferentes proteínas AGO. **B)** En animales, el complejo Drosha procesa los precursores de miRNAs que posteriormente son exportados al citoplasma por Exp5, donde estos pre-miRNAs son procesados por Dicer a formas madura, para unirse a complejos proteicos junto con AGO y cumplir así su función metabólica.
Axtel *et al.* 2011[11].

Biogénesis y función

Los miRNAs se originan de la transcripción de regiones no codificantes del genoma, y pueden estar embebidos en regiones intrónicas de determinados genes, co-expresandose conjuntamente con ellos[12,13].

Inicialmente, los miRNAs son transcritos a partir de la acción de la ARN polimerasa II (RNApol II)[14], generándose transcritos primarios de miRNAs (pri-miRNAs), consistentes en secuencias de varias Kilobases de longitud[15], que posteriormente son reconocidas por la enzima DGCR8, también denominada Pasha[16], asociada a la enzima RNAse-III Drosha[17], en un complejo proteico denominado *Microprocessor complex*[18], que actúa sobre la secuencia del pri-miRNA para formar estructuras de ARN monocatenario de entre 50 a 150 nucleótidos aproximadamente, denominadas precursores de miRNAs (pre-miRNAs), y consistentes en cadenas plegadas sobre sí mismas en forma de horquilla, con un *loop* o giro terminal y dos brazos o *stems*, unidos de forma complementaria más o menos imperfectamente.

Posteriormente estos pre-miRNAs son exportados desde el núcleo donde se han generado, hacia el citoplasma celular[19], atravesando la membrana celular a través de poros nucleares mediante la interacción con el complejo formado por la proteína *Exportin-5 (*XPO5) y Ran-GTP[20,21].

Una vez en el citoplasma, los pre-miRNAs son reconocidos y procesados por la enzima RNAse-III Dicer[22], que reconoce ambos brazos o *stems* del pre-miRNA y corta dicha estructura[23], eliminando el giro terminal y generando una estructura de RNA bicatenario de entre 18 a 24 nucleótidos que conforma el complejo miRNA:miRNA*. Seguidamente, una de las cadenas de este complejo intermedio de miRNA maduro será degradada (cadena * o *passenger*) por la acción de helicasas y ARNasas[24], mientras que la otra (cadena guía), será conservada e introducida en el denominado *RNA-induced silencing complex* (RISC), conjuntamente con proteínas endonucleasas Argonauta 2 (Ago 2)[25], mediante la acción de la proteína TRBP[26], donde cumplirá su función metabólica como miRNA maduro. Esta función se centra en la capacidad del complejo RISC, donde los miRNAs están embebidos, para unirse a mRNAs diana mediante la interacción específica (aunque existen múltiples variantes en esta interacción[27]) entre la región *seed* del miRNA maduro , formada por los 6-8 nucleótidos iniciales de la región 5’ del miRNA[28,29], y la región 3’-UTR de los mRNAs diana[30,31], reduciendo o dificultando, de este modo, el proceso de translación o procesado del mRNA por parte de los ribosomas, o bien induciendo la degradación del propio mRNA diana mediante el incremento de la velocidad de acortamiento de la cadena de poli-A de los transcritos de ARN[32].

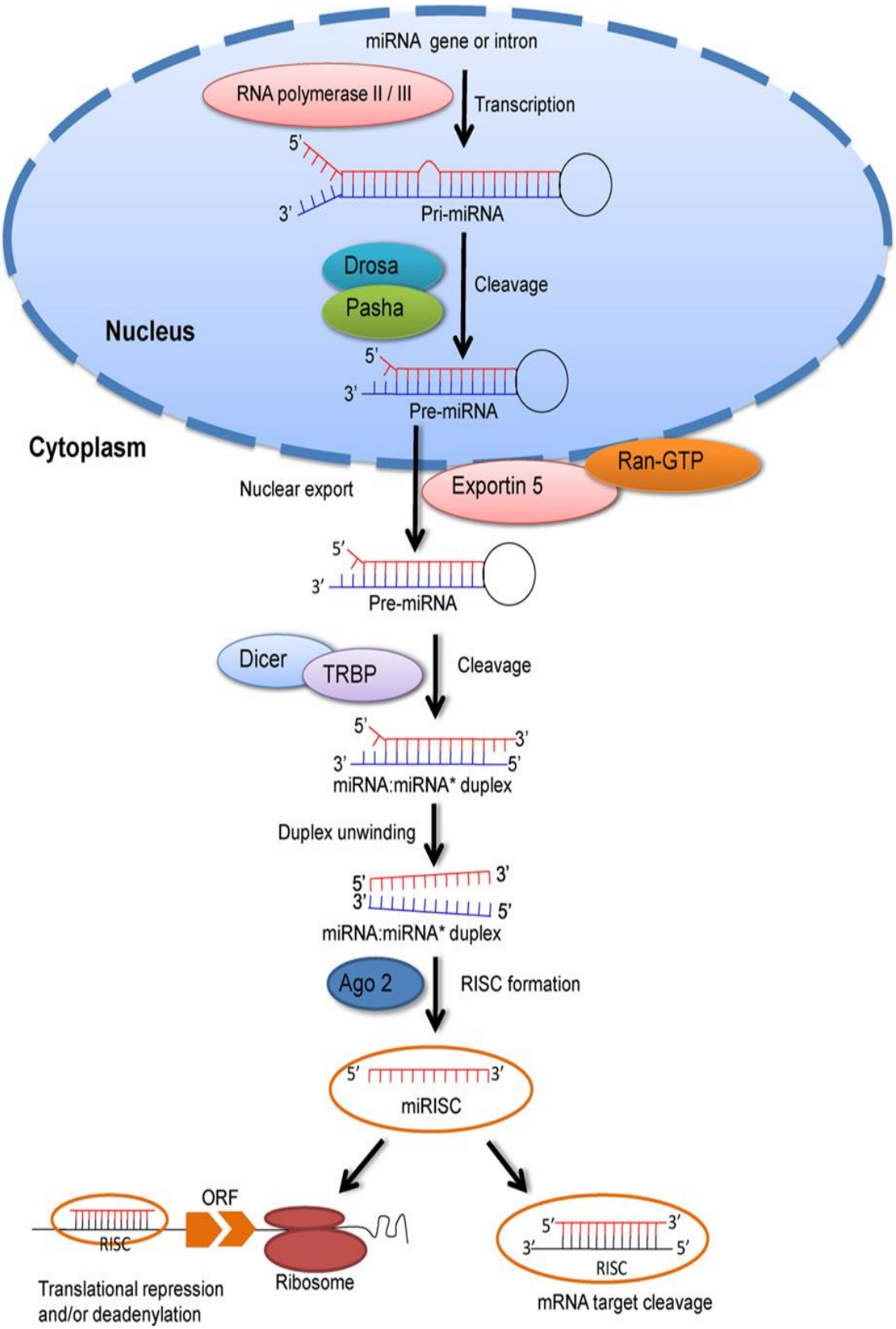

Figura 3: Biogénesis y función de los microRNAs. Modificado de Akhtar *et al*. 2016[33].

Mecanismos menores alternativos de acción de los miRNAs puede incluir la unión de éstos a determinadas proteínas de unión a ARN (*RNA-binding proteins*), dificultando de este modo su interacción y unión con ARNs diana[34], o bien actuando directamente sobre la expresión génica a nivel transcripcional, uniéndose directamente a proteínas reguladoras de la transcripción del ADN[35]. Por otra parte, también se han podido identificar miRNAs con funciones de activación de la transcripción[36].

Conservación filogenética

Desde los inicios del estudio sobre la naturaleza y función de los miRNAs[37–39], se detectó la presencia de secuencias homólogas en diferentes especies[40,41], lo que les convirtió en marcadores moleculares interesantes para el análisis de la variación y conservación evolutiva durante el proceso de especiación, debido a su reducida tasa de variación[42], partiendo de la asunción de que una posible conservación filogenética, estaría probablemente ligada a una conservación biológica funcional. No obstante, algunos estudios han puesto en tela de juicio la extensión de ciertos consensos hacia una concepción universal de los microRNAs como estructuras estables y conservadas a lo largo de la historia evolutiva[43].

En este sentido podemos destacar a la microRNA *let-7*, que posee una extensa conservación entre diferentes especies analizadas[44,45]. Estudios realizados en *C. elegans*, han podido relacionar la actividad de *let-7* con la transición desde estados larvarios a adultos[46], función análoga a la descrita para el caso de la regulación del proceso de metamorfosis en *D. melanogaster*, guiando la transición desde larva a fases adultas[47]. Para el caso de especies vertebradas, let-7 también ha podido ser identificado como uno de los responsables en cambios durante el desarrollo embrionario en peces[48,49], mientras que en el caso de vertebrados superiores, se han podido identificar numerosas dianas relacionadas con procesos de desarrollo y diferenciación celular tejido-específicos[50].

Otro ejemplo podemos encontrarlo en el microRNA 1 (*miR-1*), uno de los miRNAs más importantes y mayoritariamente expresados en miocitos, e íntimamente relacionado con el proceso de cardiogénesis y desarrollo de tejido muscular esquelético[51]. Este miRNA está regulado en su expresión por los factores de transcripción *Mef-2* y *MyoD*[52], hecho que también ha sido posible detectar en el caso de *D. melanogaster*, lo que sugeriría una conservación en el proceso de regulación de la función y expresión de *miR-1* entre insectos y vertebrados[53]. No obstante, y pese a la amplia conservación de la expresión y regulación del circuito metabólico de *miR-1*, se ha podido determinar una gran variabilidad en cuanto al rango de mRNAs diana influenciados por su acción represora[54], abundando en el hecho de la manifiesta conservación evolutiva en la secuencia y expresión de numerosos miRNAs, no así, sin embargo, si tenemos en cuenta su interacción con mRNAs dianas a los cuales regulan en su expresión, donde existe una mucho mayor variabilidad evolutiva.

Estos resultados inducen a determinar el gran peso de *let-7* y *miR-1*, así como otros miRNAs, en numerosos procesos biológicos de gran interés para su estudio como son el desarrollo y diferenciación celular o procesos relacionados con el cáncer. Pese a ello, es importante recalcar la gran variabilidad y complejidad de los numerosos y muy interconectados circuitos metabólicos en los cuales los miRNAs pueden cumplir un papel principal.

Detección de miRNAs

La detección y anotación de miRNAs ha supuesto y supone en la actualidad, ciertamente un desafío en cuanto a la correcta identificación y caracterización de las secuencias compatibles con estructuras biológicamente asimilables a microRNAs. Estos esfuerzos han derivado en la identificación hasta la fecha de varios miles de miRNAs en la totalidad de especies analizadas, pese a que el número de secuencias aún desconocidas es seguramente mucho mayor aún[55]. Las nuevas técnicas de secuenciación *high-throughput* desarrolladas en los últimos años han contribuido decididamente a incrementar el catálogo de miRNAs descritos en multitud de especies animales, plantas y virus[56,57], siendo almacenadas en bases de datos para su consulta y acceso público. Una de las bases de datos de referencia para miRNAs es miRBase (http//:www.mirbase.org), que en su última versión (*Release 22*) de marzo de 2018, incluye un total de 38.589 secuencias precursoras de miRNAs, y 48.885 secuencias de miRNAs maduros anotados[58].

Son numerosas las aproximaciones empleadas para la detección de miRNAs, desde las técnicas clásicas de clonación[55] o hibridación *in situ*[59], hasta las más modernas basadas en técnicas de NGS[60]. Fundamentalmente estas técnicas se han centrado en la comparación de la homología entre especies, haciendo uso de la característica conservación filogenética mostrada por los miRNAs[42], así como en el análisis de características estructurales específicas de los miRNAs, como sus energías libres de plegamiento o la distribución de tipos de enlace entre bases nitrogenadas[61]. Para ello, se han desarrollado programas que realizan estimaciones del plegamiento idóneo de las secuencias de RNA monocatenario, así como cálculos de las energías libres necesarias para ello, como son RNAfold[62] o UNAFold[63].

Sistemas por homología:

Los sistemas de detección y anotación de miRNAs basados en homología, basan su principio en la búsqueda de secuencias homólogas entre especies diferentes, asumiendo la posibilidad de que un mismo microRNA identificado en algún organismo, posea su contraparte homóloga en otro organismo en estudio[64]. Esta aproximación puede resultar útil sobre todo en la anotación de organismos cuyos genomas han sido aún poco estudiados y cuya calidad y fiabilidad de anotación sea incipiente[65]. Ejemplos de esta aproximación son diferentes herramientas que utilizan técnicas semejantes al alineamiento BLAST, como MiRscan[39], miRseeker[66], RNAmicro[67], mirOrtho[68] o proMiR II[69].

No obstante, este tipo de aproximaciones han demostrado resultados discretos en cuanto a la validación funcional de microRNAs, pese a detectar un elevado número de secuencias homólogas conservadas presumiblemente compatibles con la estructura de miRNAs[70], además de estar limitadas a la caracterización de nuevas secuencias con homólogos descritos, obviando la posible y previsible existencia de miRNAs especie-específicos aún no detectados. Por otra parte, la identificación de homologías entre miRNAs que han sufrido gran cantidad de variaciones en sus secuencias debido a rápidos fenómenos de especiación o divergencia evolutiva, pueden ser muy difíciles de detectar y validar mediante estos métodos.

Sistemas *ab-initio*:

Al contrario que las aproximaciones basadas en homología, este tipo de aproximaciones tienen la ventaja de no necesitar la anotación previa de una correspondiente estructura homóloga para su identificación, lo que aporta la posibilidad de detectar miRNAs especie-específicos o cuyos homólogos aún no hayan sido caracterizados convenientemente[71].

Uno de los puntos clave en el desarrollo de este tipo de modelos de predicción de miRNAs, estriba en la correcta selección de atributos para caracterizar las secuencias compatibles con estructuras de miRNAs, de aquellas otras que no lo son. Fundamentalmente, la mayoría de modelos desarrollados en este sentido hacen uso de características estructurales derivadas del plegamiento de las formas precursoras de miRNAs (pre-miRNAs), como son la energía libre de plegamiento y otros valores derivados[72]. Este tipo de recursos estructurales puede derivar en problemas en el caso de que los atributos calculados no definan de forma totalmente inequívoca a los miRNAs respecto de otras estructuras, dando lugar a una elevada tasa de falsos positivos, con especificidades bajas en la detección de miRNAs. El caso contrario puede suceder cuando los atributos calculados no sean suficientes para clasificar determinadas estructuras como miRNAs, dando lugar a elevadas tasas de falsos negativos, y sensibilidades bajas.

Para llevar a cabo el proceso de predicción ab-initio, en los últimos años se han desarrollado numerosas herramientas basadas en técnicas de *Machine Learning*, utilizando diversos algoritmos de aprendizaje automático como *Support Vector Machines* (SVMs)[68,73,74], *hidden Markov Models* (HMM)[75–78], *hierarchical Random Forests*[79,80] o clasificadores *naïve Bayes*[81], entre otros. Resulta particularmente importante la definición de toda una serie de atributos que actuarán como variables para la detección y clasificación de las estructuras precursoras de miRNAs (pre-miRNAs), así como seleccionar sets de datos de partida positivos y negativos para entrenar los modelos eficazmente[82,83], siendo esta una fuente habitual de conflicto sobre todo en especies cuyos genomas no han sido aún secuenciados o cuyas anotaciones son particularmente escasas e ineficientes.

Datos positivos:

Como datos positivos seleccionaremos todas aquellas secuencias cuya anotación se corresponda con microRNAs en la especie en la que estemos interesados. Normalmente, para el desarrollo del set de datos de secuencias positivas, se puede hacer uso de la base de datos miRBase[58], considerada como una de las bases de datos de referencia para el acceso y consulta de anotaciones de microRNAs. Para el caso del porcino, posee, en su última versión actualizada, alrededor de 300 miRNAs anotados. Debemos, no obstante tener precaución en cuanto a la fiabilidad de la anotación de ciertas estructuras de miRNAs en la base de datos miRBase[83], debido a la poca fiabilidad de las predicciones de plegamiento y límites asumidos de miRNAs maduros funcionales para ciertas anotaciones, así como a la presencia de estructuras sin anotación espacial en el genoma, pese a encontrarse anotadas como tales en la base de datos Ensembl[84]. Existen, no obstante, otras bases de datos de miRNAs de reciente publicación, con anotaciones más fiables[85], pero aún con información incompleta o ausente respecto a organismos no modelo como puede ser el porcino. Para solventar esto, se puede hacer uso de la anotación en bases de datos como miRTarBase[86], que proveen interacciones miRNA-mRNA descritas de forma experimental, asumiendo también el hándicap de la escasa anotación experimental para multitud de miRNAs aún no convenientemente caracterizados, sobre todo en especies no modelo.

Datos negativos:

Como datos potenciales a considerar como negativos, podremos englobar todas aquellas secuencias no anotadas como microRNAs. Esto supone, no obstante, un reto añadido al hecho de intentar identificar secuencias que bajo ningún concepto interactuarán como miRNAs para la maquinaria enzimática de maduración y que nunca derivarán en estructuras que actúen metabólicamente como miRNAs. Dada la gran variabilidad de secuencias que son candidatas a considerar para seleccionar como datos negativos, deberemos tener la precaución de elegir aquellas que sean lo suficientemente diferenciadas de las características estructurales de los miRNAs, pues corremos el riesgo de que el modelo de predicción sea incapaz de discernir entre secuencias positivas y negativas; y, al contrario, deberemos procurar que las secuencias negativas seleccionadas no sean demasiado artificiales, en caso de ser generadas de forma aleatoria o mediante modificaciones en las secuencias positivas, puesto que en tal caso, el modelo podría no ser entrenado de forma adecuada.

Una de las posibles aproximaciones, es el generar un set de datos aleatorio de secuencias artificiales o a partir de secuencias exónicas[82,87], dado que los microRNAs se alojan en regiones intrónicas del genoma[88], aunque algunos trabajos en humano[89] han utilizado selecciones aleatorias de pseudo-estructuras en horquilla filtradas desde la base de datos RefSeq[90].

Por otra parte, será importante tener en cuenta el equilibrio de casos entre los datos positivos y negativos seleccionados, aplicando estrategias para intentar minimizar el problema conocido como *class-imbalance* en el entrenamiento de algoritmos de *Machine Learning* supervisado[91]. En el caso de los microRNAs, el problema del *class-imbalance* resulta particularmente

interesante, puesto que frente a un set de cientos o pocos miles de miRNAs identificados para cada especie, que podremos a priori considerar como datos positivos a utilizar, podríamos considerar el resto de secuencias no anotadas como miRNAs, como datos negativos, suponiendo un orden de desequilibrio entre sets de datos positivo y negativo que obligatoriamente deberemos tratar de minimizar, con el objetivo de entrenar el modelo de forma correcta. Para ello, se han descrito numerosas estrategias[91]. Una de las más intuitivas, para el caso que nos ocupa respecto a la detección de microRNAs, es la del muestreo aleatorio de secuencias negativas (no anotadas como miRNAs), también conocido como *under-sampling*, hasta alcanzar cierto equilibrio entre el set de datos positivo y negativo[92], con el inconveniente de poder estar eliminando información valiosa para el proceso de clasificación durante el muestreo. Otras aproximaciones pueden basarse, por ejemplo, en utilizar determinados algoritmos para aumentar el set de datos positivos, generando nuevas unidades sintéticas para la clase minoritaria[93], o en muestreos basados en selección de atributos[94].

Por otra parte, es también posible hacer uso de modelos semi-supervisados aplicando estrategias de clusterización de datos no clasificados respecto a muestras positivas o negativas, con el objetivo de construir los sets de datos para entrenar los algoritmos de *Machine Learning*[95], estrategia que ya se ha demostrado recientemente eficaz en la predicción a nivel genómico de estructuras compatibles con precursores de microRNAs[96].

Representación de secuencias miRNA

Definidos los diferentes modelos disponibles para realizar el proceso de clasificación de secuencias microRNA, así como las posibles estrategias a seguir y los sets de datos de partida para el entrenamiento de algoritmos basados en *Machine Learning*, resulta necesario transformar las secuencias génicas de partida, formadas por cadenas de nucleótidos representados por las letras A,U,C,G, símbolo de las bases nitrogenadas que contienen, en elementos que puedan ser procesables por los diferentes algoritmos de clasificación, a modo de atributos estructurales capaces de caracterizar las secuencias analizadas, , y por lo tanto contribuir al proceso de clasificación por parte de los algoritmos de *Machine Learning*.

Uno de los primeros trabajos que implementó la extracción de atributos estructurales a partir de secuencias fue el reportado por Pfeffer et al. 2005[97], en el que se definieron por primera vez algunos de los atributos que posteriormente se han ido utilizando en trabajos posteriores. Estos atributos iniciales fueron, entre otros, la energía mínima libre de plegamiento (*mínimum free energy*), normalmente obtenida a partir de softwares de estimación de plegamiento como RNAfold[62] o UNAFold[63], longitud de los brazos o *stems* de la cadena plegada, el contenido en cada una de las bases nitrogenadas de la secuencia, o la proporción de enlaces A-U y G-C, así como el enlace de tipo *wobble* G-U, que no sigue las reglas de enlaces tipo Watson-Crick. Otros trabajos como el desarrollado por Ng *et al*. 2007[98], ampliaron los atributos

estructurales calculados añadiendo atributos de configuración de secuencia a partir de las frecuencias dinucleotídicas, tal que XY%, donde X,Y∈{A,C,G,U}, además de otros atributos de plegamiento como la predisposición de enlace entre bases (*base-pair Propensity*)[99], energía mínima libre de plegamiento ajustada por longitud de la secuencia (*adjusted MFE*)[100,101], distancia de enlace entre bases ajustada (*adjusted base-pair Distance*)[101,102] o la entropía de Shannon ajustada (*adjusted Shannon Entropy*)[101] y atributos topológicos como el grado de compactación (*degree of Compactness*)[103], entre otros.

Otros ejemplos fueron los desarrollados por Xue *et al*. 2005[87], que contribuyeron a ampliar el catálogo de atributos con su estimación de elementos tripletes, para representar la configuración contigua estructural de secuencia, o los propuestos por Batuwita *et al*. 2009[74], que introdujeron el cálculo de atributos como la energía libre de ensamblado ajustada (*adjusted ensemble free energy*), frecuencia de la estructura MFE (*MFE Frequency*), diversidad estructural (*structural Diversity*), entropía estructural (*structural Entropy*), entalpía estructural (*structural Enthalpy*), energía de fusión de estructura (*Melting structure energy*), o las frecuencias dinucleotídicas normalizadas por longitud de la secuencia.

Todos estos atributos han sido desde entonces utilizados de forma asidua por las diferentes aproximaciones computacionales para la predicción de microRNAs, integrándolas en mayor o menor medida en programas de cálculo a partir de secuencias a evaluar. Un ejemplo de ello puede encontrarse en el trabajo reportado por Yones *et al*. 2015, mediante el software *miRNAfe*[104], que integra la mayoría de los atributos anteriormente citados por otros trabajos de predicción en un entorno de lenguaje MATLAB para la extracción de atributos estructurales a partir de secuencias.

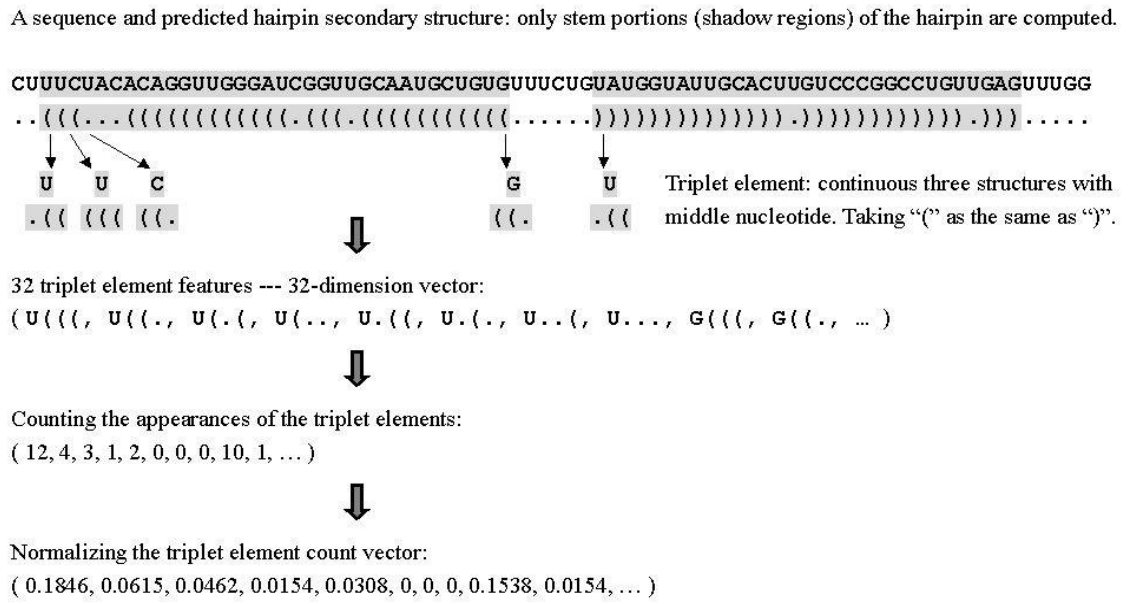

Figura 4: Ejemplo del cálculo de tripletes para representar la estructura de secuencia local en el pre-miRNA. Cada triplete está compuesto por tres subestructuras continuas y el tipo de nucleótido central. La posible aparición de cada una de las 32 combinaciones de tripletes posibles a lo largo del pre-miRNA son representadas en un vector normalizado de 32 unidades.

1.2. Justificación del TFM

A nivel nacional, el sector de la ganadería de producción porcina se sitúa en el primer puesto en facturación, con alrededor de 4.000 millones de euros anuales, lo que sitúa a España como el segundo productor en importancia de la Unión Europea[105]. Por otra parte, las proyecciones económicas sitúan al consumo y producción mundial de carne de porcino en segunda posición en importancia, con un incremento estimado de alrededor del 8% para 2026[106]. La caracterización y anotación de estructuras génicas codificantes y no codificantes en la actual versión del genoma secuenciado en esta especie, es aún incipiente, existiendo solamente alrededor de 300 miRNAs anotados en la última versión actualizada de la base de datos de referencia miRBase[58], y sólo 36 interacciones miRNA-mRNA experimentalmente descritas en la base de datos miRTarBase[86].

Se pretende profundizar acerca del desempeño computacional de los algoritmos de predicción de miRNAs basados en *Machine Learning*[107], aplicados al caso del porcino, así como el desarrollo de nuevas aproximaciones que puedan mejorar la sensibilidad y especificidad de los algoritmos predictores y clasificadores, lo que presumiblemente supondrá extender y mejorar la anotación de miRNAs en dicha especie, redundando en un beneficio para futuros trabajos en transcriptómica y regulación génica en porcino.

1.3. Objetivos del Trabajo

Objetivos generales

Los objetivos generales del proyecto de Trabajo Fin de Máster son los siguientes:

1. Evaluación del rendimiento de diferentes algoritmos de predicción de microRNAs en el genoma porcino basados en reconocimiento por homología y en algoritmos de *Machine Learning*.

2. Predicción de nuevas estructuras compatibles con microRNAs en el genoma porcino, utilizando nuevas aproximaciones basadas en técnicas de *Machine Learning* actuales y comparar su rendimiento con otras previas.

3. Análisis de las funciones biológicas y rutas metabólicas involucradas respecto a nuevos microRNAs anotados.

Objetivos específicos

Se detallan a continuación los objetivos específicos derivados de los objetivos generales anteriormente descritos:

1. Investigación bibliográfica acerca de algoritmos desarrollados preexistentes para la predicción de microRNAs en mamíferos mediante técnicas de reconocimiento por homología, técnicas de clasificación supervisada de *Machine Learning* y nuevas aproximaciones basadas en algoritmos de *Machine Learning* del estado del arte.

2. Recolección y elaboración de sets de datos positivos y negativos para el entrenamiento de los modelos en predicción de microRNAs en porcino.

3. Evaluación del rendimiento de algoritmos de *Machine Learning* seleccionados para la predicción de microRNAs en el genoma porcino.

4. Comparación del grado de homología de los resultados obtenidos en porcino respecto a la más extensa anotación de microRNAs en humano.

5. Análisis de la funcionalidad biológica de las estructuras de microRNA predichas en porcino.

1.3. Metodología

Teniendo en cuenta la escasa y poco confiable anotación de estructuras compatibles con microRNAs en la última versión del genoma porcino (https://www.ncbi.nlm.nih.gov/genome/annotation_euk/Sus_scrofa/106/), resulta evidente que una de las tareas más importantes en cuanto al desarrollo de este trabajo, será la construcción de una base de datos de secuencias de microRNAs positivas, es decir, secuencias que muestran una elevada confianza para ser consideradas como miRNAs; y negativas, formada por secuencias que no pueden en ningún caso ser reconocidas como miRNAs, por no adquirir la estructura secundaria adecuada, así como por no poder ser procesadas por el complejo enzimático *Dicer/Drosha* durante la maduración desde estructuras precursoras a maduras funcionales. Para ello, haremos uso de la anotación de miRNAs de porcino almacenada en la base de datos de ARNs no codificantes de Ensembl[84], para la versión 11.1 del *assembly* de *Sus scrofa*, que emplearemos como set de datos positivos. Haremos uso de esta base de datos, y no de otras disponibles como las secuencias almacenadas en miRBase[58], debido a que en su última versión disponible (v22), la anotación se corresponde con el *assembly* 10.2 para *Sus scrofa*, versión que ha sido sustituida por otra posterior y con la que hemos planteado el enfoque a seguir.

Respecto al set de datos negativos, utilizaremos, así mismo, el set de datos de secuencias anotadas de ARNs no codificantes disponibles en la base de datos Ensembl[84], seleccionado en este caso, todas aquellas secuencias no

anotadas como microRNAs. Por otra parte, se podrá evaluar la inclusión de secuencias generadas de forma artificial en el set de datos negativos, dada la limitada anotación disponible en porcino.

Una vez hayamos desarrollado los sets de datos de entrenamiento de partida, deberemos definir y caracterizar las secuencias seleccionadas mediante una serie de atributos estructurales y estadísticos, que serán incorporados a los algoritmos de *Machine Learning* para poder ser entrenados y construir el modelo clasificador que emplearemos para la detección de miRNAs.

Una vez entrenado el modelo, podremos optar por dos enfoques diferentes:

1. Predicción por homología:

   Haciendo uso de la más extensa y confiable anotación de miRNAs disponible para *Homo sapiens*, y tomando en consideración la elevada tasa de conservación filogenética que muestran los miRNAs entre especies[42], podremos utilizar la anotación de secuencias disponibles en humano, para alinearlas con el *assembly* porcino en su versión 11.1, tras lo cual obtendremos una estimación para las posiciones de los miRNAs anotados en humano, en el genoma porcino, las cuales utilizaremos para reconstruir estructuras pre-miRNA elongando las secuencias maduras alineadas, y posteriormente extraeremos los atributos estructurales definidos en el entrenamiento de los algoritmos de *Machine Learning*, y procederemos finalmente a clasificar dichas secuencias como miRNAs o no. Con este enfoque nos centraremos en investigar aquellos miRNAs ya anotados en humano, para localizar sus correspondientes homólogos en porcino, anotados o no, por lo que limitaremos la búsqueda a secuencias de miRNAs conservadas y previamente anotadas en la especie modelo de referencia, obviando la búsqueda para secuencias miRNA especie-específicas no anotadas en porcino, o cuya estructura homóloga no haya sido detectada y anotada en la especie humana.

   Pese a la evidente limitación de esta estrategia, dada la aún limitada y escasa anotación de miRNAs en porcino, podemos tomarla como un punto de partida interesante para la detección de estructuras homólogas de miRNAs bien caracterizadas y descritas en humano, que, pese a ello, aún no han sido anotadas en el genoma porcino.

2. Predicción *ab-initio*:

   Otra aproximación que podemos considerar es la predicción directa de secuencias de miRNA procedentes de la especie analizada, en nuestro caso, el porcino. Para este enfoque, se ha planteado el uso de datos de secuenciación de *small-RNAseq* disponibles en porcino en repositorios públicos (https://www.ncbi.nlm.nih.gov/sra/). Estos archivos de secuenciación pueden ser utilizados para predecir estructuras compatibles con microRNAs, en una forma similar a la utilizada en la predicción por homología. Inicialmente deberemos descargar y transformar los archivos SRA a ficheros en formato FASTQ, para a

continuación colapsar dichos ficheros de secuenciación con el fin de eliminar las secuencias repetitivas. Extraeremos las secuencias y procederemos a su alineamiento y posterior filtrado y elongación para reconstruir estructuras pre-miRNA, que utilizaremos como input al modelo de predicción entrenado previamente, con el fin de clasificar dichas secuencias como miRNAs o no.

Esta aproximación nos permitirá trabajar directamente con secuencias de ARNs pequeños expresados en porcino, según sea la procedencia experimental del análisis de secuenciación del que hayamos extraído dichas secuencias, obviando por tanto la necesidad de utilizar una especie modelo como puede ser la anotación de miRNAs en humano. La principal limitación de este enfoque radica en la elevada cantidad de secuencias a analizar, y el elevado número de secuencias que presumiblemente serían predichas por el modelo como compatibles con una estructura de pre-miRNA, por lo que sería necesario posteriormente realizar un análisis pormenorizado de las secuencias obtenidas para detectar aquellas ya anotadas y aquellas que aún no lo estuvieran, así como plantear experimentos de detección confirmatorios mediante clonación por RT-qPCR[108].

Finalmente, dependiendo de los resultados obtenidos y la capacidad de predicción de los algoritmos empleados para el caso del porcino, teniendo en cuenta todas las particularidades ya mencionadas, podremos extender el trabajo de predicción hacia la identificación de rutas metabólicas y *targets* mRNA implicados en la funcionalidad de los nuevos microRNAs anotados[109], contrastando dicha información con la anotación en otras especies modelo como humano o ratón, donde la disponibilidad de datos de interacción miRNA-mRNA es mucho más extensa[85,86,110,111].

1.4. Planificación del Trabajo

Tareas

Del conjunto de objetivos anteriormente propuestos, se desglosa a continuación la lista de tareas aplicables realizadas:

1. Investigación bibliográfica:

a Recabar información sobre algoritmos desarrollados para la predicción de microRNAs en mamíferos mediante técnicas de reconocimiento por homología y técnicas de *Machine Learning.*

b Investigar sobre nuevas aproximaciones en la predicción de microRNAs.

2. Elaboración del set de datos de entrenamiento de los modelos:

a Evaluar la fiabilidad de la anotación de microRNAs para el genoma porcino en las bases de datos de miRBase[58] y Ensembl[84]. Elaborar set de datos positivos para porcino y humano.

b Desarrollar un set de datos de secuencias negativas mediante la generación de secuencias pseudo-miRNAs y/o a partir de secuencias de ARN no codificante anotadas en porcino y humano. Extraer datos de secuencias negativas de trabajos previos de predicción y clasificación de miRNAs[98].

3. Desarrollo de algoritmo de extracción de atributos estructurales de secuencias pre-miRNA:

a Generar un algoritmo en código R para filtrar y extraer atributos estructurales a partir de secuencias pre-miRNA.

b Extraer atributos estructurales a partir del set de datos positivo y negativo.

4. Evaluar y contrastar el rendimiento de diferentes algoritmos de predicción de microRNAs en el genoma porcino:

a Seleccionar una serie de algoritmos de clasificación de *Machine Learning*, y evaluar su rendimiento con el set de datos de entrenamiento. Contrastar el rendimiento de predicción del mejor modelo respecto a otros datos de entrenamiento en especies modelo o publicados en otros trabajos previos.

b Clasificar secuencias pre-miRNA a partir de una base de datos input de secuencias generada mediante un análisis por homología utilizando la anotación de miRNAs de humano.

5. Analizar el grado de homología de las predicciones de microRNA respecto a la anotación en especies modelo:

a Seleccionar aquellos microRNAs nuevos no anotados previamente en porcino, que muestren una mayor confianza de clasificación.

b Contrastar su secuencia respecto a la anotación de microRNAs en humano y ratón.

6. Análisis funcional de microRNAs predichos:

a Analizar la posible anotación funcional de microRNAs seleccionados que presenten algún grado de homología respecto a microRNAs anotados y funcionalmente caracterizados en especies modelo.

b Analizar de forma preliminar la posible detección de dianas en regiones 3'-UTR de ARN mensajeros (ARNm) diana.

Calendarización

Tarea	Fecha Inicio	Fecha Fin
PEC0 – Definición de Contenidos	21/02/2018	05/03/2018
PEC1 – Plan de Trabajo	06/03/2018	19/03/2018
PEC2 – Desarrollo del Trabajo. Fase 1	20/03/2018	23/04/2018
❖ Investigación bibliográfica	20/03/2018	01/04/2018
• Homología y *Machine Learning*	20/03/2018	25/03/2018
• Nuevas técnicas en *Machine Learning*	26/03/2018	01/04/2018
❖ Datos de entrenamiento	02/04/2018	10/04/2018
• Set de datos positivos	02/04/2018	05/04/2018
• Set de datos negativos	06/04/2018	10/04/2018
❖ Rendimiento de algoritmos	11/04/2018	23/04/2018
• Extracción de atributos de pre-miRNAs	11/04/2018	22/04/2018
• Selección de algoritmos	22/04/2018	23/04/2018
PEC3 – Desarrollo del Trabajo. Fase 2	24/04/2018	21/05/2018
• Entrenamiento del clasificador	24/04/2018	06/05/2018
• Predicción por homología	07/05/2018	13/05/2018
❖ Homología de las predicciones	14/05/2018	21/05/2018
• Selección de miRNAs	14/05/2018	15/05/2018
• Contraste con especies modelo	15/05/2018	21/05/2018
❖ Análisis funcional	16/05/2018	21/05/2018
• Función por homología	17/05/2018	21/05/2018
• Análisis de ARNm dianas	17/05/2018	21/05/2018
PEC4 – Redacción de la Memoria	20/03/2018	05/06/2018
PEC5a – Elaboración de la Presentación	06/06/2018	13/06/2018
PEC5b – Defensa pública	14/06/2018	25/06/2018

Tabla 1: Calendarización propuesta.

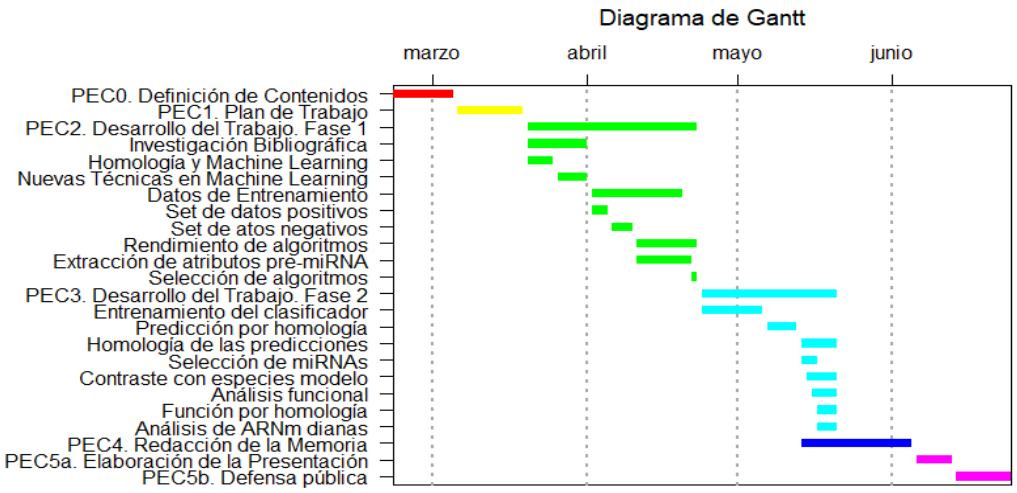

Figura 5: Calendarización mediante Diagrama de Gantt.

Hitos

Hitos marcados por el plan docente del Trabajo de Fin de Máster:

PEC1: Se hace entrega de un plan de trabajo donde se definen las líneas generales del proyecto, el enfoque a aplicar, los objetivos y tareas necesarias para el cumplimiento de dichos objetivos, así como la organización temporal de todo el proyecto.

PEC2: Se hace entrega de un informe evaluando el desarrollo del proyecto hasta el momento, indicando y justificando los cambios organizativos que se pudieran producir. Revisar la temporización prevista e indicar y justificar posibles desviaciones respecto al proyecto original si procede, proponiendo acciones de mitigación.

PEC3: Se hace entrega de un informe evaluando el avance del proyecto respecto a la anterior entrega, justificando los cambios realizados y detallando las actividades realizadas, previstas y no previstas. Además, se detallarán los resultados obtenidos.

PEC4: Se deposita la memoria del Trabajo de Fin de Máster.

PEC5a: Se entrega la presentación del Trabajo Fin de Máster, que sintetice de forma clara y concisa el trabajo realizado y los resultados obtenidos, ofreciendo una perspectiva general del proyecto y recogiendo los aspectos más importantes del mismo. Constará de 20 páginas o transparencias y se presentará de forma oral en un tiempo máximo de 20 minutos.

Hitos relacionados con el cumplimiento de las tareas:

1. Se realiza entrega del set de datos de entrenamiento construido para su evaluación. 20/04/2018

2. Se realiza entrega de informe de selección de algoritmos a evaluar para contrastar su rendimiento respecto a nuevas aproximaciones en desarrollo. 23/04/2018

3. Se realiza entrega de informe con resultados obtenidos en la predicción de miRNAs, código utilizado y análisis de funcionalidad biológica e interacciones miRNA-mRNA en microRNAs predichos seleccionados. 21/05/2018.

Desviaciones respecto al Plan de Trabajo inicial

Se destacan a continuación las principales desviaciones y aportaciones realizadas respecto al Plan de Trabajo inicial:

1. Debido a la limitación en cuanto a anotación de secuencias en la especie porcina, se decidió centrar la revisión bibliográfica y las consideraciones metodológicas en el uso de algoritmos de *Machine Learning* basados en aprendizaje supervisado utilizando sets de datos positivos y negativos, atendiendo a limitar el efecto del class-imbalance. Otras aproximaciones basadas en aprendizaje semi-supervisado han sido aplicadas con éxito[96], pero se consideró que excedían la planificación objetiva de este trabajo.

2. El escaso número de microRNAs en porcino que han sido sujetos a validación experimental mediante técnicas no *in silico*, accesibles a través de bases de datos públicas, así como el aún incipiente campo de estudio sobre sistemas de clasificación *one-class*, motivó que se decidiera basar el estudio del poder de predicción de modelos de *Machine Learning*, en algoritmos *two-class* con sets de datos positivos y negativos seleccionados a partir de bases de datos de anotación generales como miRBase[58] y Ensembl[84].

3. De los resultados obtenidos, se decidió centrar la atención en un posible microRNA no anotado hasta la fecha en porcino (ssc-miR-483), que fue detectado como homólogo de hsa-miR-483. La secuencia *seed* para miR-483-3p se demostró idéntica para ambos homólogos, pudiendo inferir *a priori* una conservación funcional respecto a los estudios realizados y el conocimiento acumulado para hsa-miR-483.

1.5. Sumario de productos obtenidos

Durante el desarrollo del presente Trabajo Fin de Máster, se han desarrollado una serie de programas encaminados a la identificación y anotación de estructuras pre-miRNAs no descritas en la especie porcina. Para ello se ha hecho uso de sets de datos positivos (1, 3) y negativos (2, 4, 5) para microRNAs en porcino y humano, así como un programa de extracción de atributos estructurales para secuencias pre-miRNA (6), desarrollado en lenguaje R. El conjunto de funciones desarrolladas en R tienen como objetivo el filtrar secuencias por tamaño y estructura, extraer y calcular una serie de atributos estructurales y estadísticos a partir de las secuencias evaluadas, y posteriormente entrenar un algoritmo basado en *Support Vector Machine* (SVM), que a continuación será utilizado para clasificar secuencias obtenidas a partir de un contraste por homología respecto a una especie modelo con mayor calidad de anotación para miRNAs, como es el humano. Para realizar el contraste por homología y obtener las secuencias a predecir, se ha generado a su vez un programa de extracción y filtrado de secuencias basado en homología (7). Una vez obtenidas las secuencias por comparación de homologías, se introducirán en el algoritmo clasificador entrenado para su predicción. Posteriormente, una vez clasificadas las secuencias seleccionadas mediante el algoritmo entrenado SVM, se ha desarrollado, así mismo, un programa para filtrar aquellas ya anotadas en la especie problema, el porcino en nuestro caso, para seleccionar las secuencias aún no anotadas y calcular un valor de confiabilidad o *Neighbouring Score*, basado en la coincidencia de elementos codificantes y no codificantes en las proximidades de la región anotada, comparando la anotación de *Homo sapiens* con la de porcino.

A continuación, se detallan los productos generados:

1. **Set de datos positivos (miRNAs) en porcino:**
   Pig_positive_set.fa
2. **Set de datos negativos en porcino:**
   Pig_negative_set.fa
3. **Set de datos positivos (miRNAs) en humano:**
   Human_positive_set.fa
4. **Set de datos negativos en humano:**
   Human_negative_set.fa
5. **Set de datos negativos pseudo-miRNAs (Ng et al. 2007):**
   Pseudo-miRNAs_set.fa
6. **Script R con funciones de filtrado, cálculo de atributos, entrenamiento y predicción de estructuras pre-miRNA:**
   eMIRNA.R
7. **Script de filtrado y extracción de secuencias por homología:**
   eMIRNA-Hunter.sh
8. **Script de filtrado y cálculo de confiabilidad de la predicción:**
   eMIRNA-Seeker y script auxiliar biomaRt_calc.R
9. **Listado de miRNAs homólogos a humano, predichos por el algoritmo SVM en el genoma porcino:**
   miRNAs_Predicted.txt

10. **Listado de miRNAs no anotados en el assembly porcino 11.1, y posiciones genómicas inferidas, así como valor de confiabilidad:** Novel_miRNAs_Predicted.txt

1.6. Otros capítulos del TFM

En los próximos capítulos de esta memoria de Trabajo de Fin de Máster se desgranarán los diferentes métodos utilizados y productos generados para el análisis y predicción de nuevas estructuras compatibles con pre-miRNAs, aún no anotadas en porcino. El capítulo 2 está dedicado por entero a desgranar cada uno de los pasos secuenciales para finalmente obtener un listado de posibles candidatos a nuevos microRNAs no anotados previamente en porcino, detallando sus posiciones inferidas mediante el contraste por homología, así como sus valores de confiabilidad para ser tenidos en cuenta o no como candidatos sólidos para una posterior validación a nivel molecular mediante técnicas de RT-qPCR. El capítulo 3 detalla los resultados obtenidos y la discusión crítica de los mismos, con los hallazgos más interesantes encontrados tras completar todo el proceso bioinformático. Finalmente, el capítulo 4 enumera las principales conclusiones obtenidas de la realización de este proyecto de Trabajo Fin de Máster, y posibles aplicaciones futuras.

2. Material y métodos

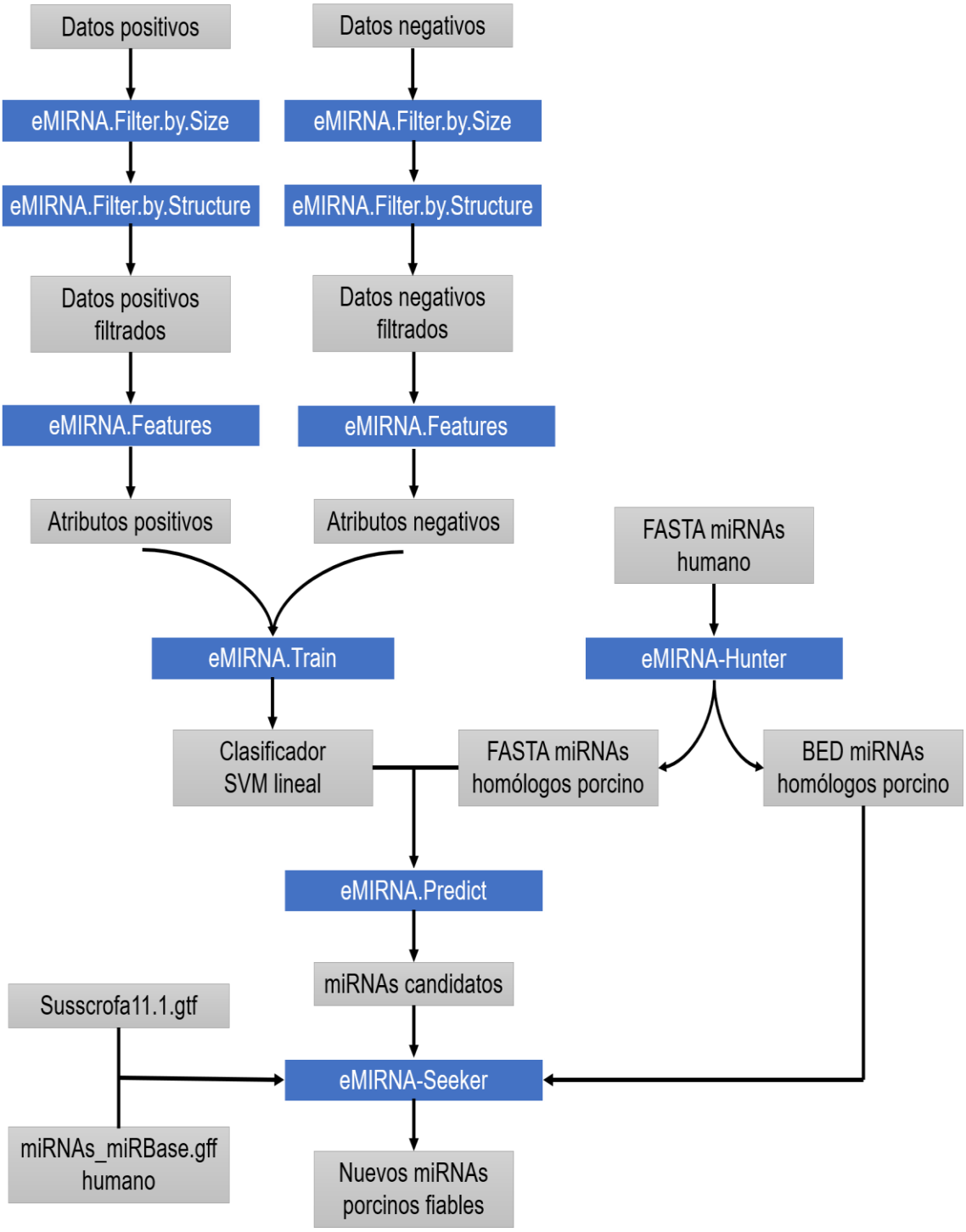

Figura 6: Esquema del procedimiento seguido desde la generación de los sets de datos positivos y negativos hasta la obtención de miRNAs candidatos no anotados en el *assembly* porcino 11.1 fiables de acuerdo con el *Neighbouring Score* calculado.

2.1. Selección de datos

El primer paso para el diseño del protocolo de detección y predicción de microRNAs nuevos en el genoma porcino, consistió en definir el set de datos positivos y negativos para su uso en el entrenamiento de algoritmos basados en *Machine Learning*, con el objetivo de entrenar un modelo de predicción para clasificar secuencias pre-miRNA, y poder así detectar nuevos candidatos a miRNAs no anotados en porcino.

Para la obtención del set de datos positivo y negativo, se optó por descargar la anotación de ARNs no codificantes para el *assembly* porcino en su versión 11.1, disponible en la base de datos Ensembl[84]. Decidió utilizarse esta aproximación y no la base de datos disponible en miRBase[58], debido a que para la última actualización en dicha base datos (v22), aún no puede encontrarse disponible la más reciente versión del *assembly* porcino, estando aún referenciada la anotación para microRNAs respecto a la anterior versión del genoma porcino (10.2).

Datos positivos

A partir de la anotación de secuencias de ARN no codificante para el *assembly* porcino 11.1 disponible en Ensembl (ftp://ftp.ensembl.org/pub/release-92/fasta/sus_scrofa/ncrna/), se filtraron aquellas secuencias anotadas como microRNAs, generándose un fichero en formato FASTA con un total de 554 secuencias correspondientes a estructuras pre-miRNA en porcino, englobando al total de secuencias actualmente descritas y anotadas como miRNAs en el genoma porcino, en la base de datos Ensembl.

Por otra parte, se optó además por descargar las secuencias de ARN no codificantes para humano (ftp://ftp.ensembl.org/pub/release-92/fasta/homo_sapiens/ncrna/), y se procedió a filtrar las secuencias disponibles anotadas como miRNAs, de forma análoga a lo realizado para la anotación en porcino.

Datos negativos

En lo referente al set de datos negativos, se hizo uso así mismo de las secuencias disponibles en los archivos de ARN no codificante para el *assembly* porcino y humano, no anotadas como microRNAs. Por otra parte, se decidió incluir el uso del set de datos de pseudo-miRNAs publicado por Ng *et al*. 2007[98], que comprende un total de 7185 secuencias de ARN generadas de forma aleatoria a partir de secuencias CDS disponibles en la base de datos Refseq[90].

Por otra parte, y para evitar las posibles consecuencias de un desequilibrio entre clases (*class-imbalance*), se decidió suplementar la anotación de secuencias negativas en el set de datos para porcino, mediante la generación de secuencias artificiales a partir de la modificación de los pre-

miRNAs originales correspondientes al set de datos positivos. Para ello, se hizo uso del software MutateDNA de sms2 tools[112], generando secuencias *mock-miRNA*, con una tasa de inserción de mutaciones de 50 por cada 100 nucleótidos, que fueron añadidas al set de datos negativos de ARNs no codificantes.

Para el resto de sets de datos positivos y negativos evaluados, no se consideró necesaria la inclusión o exclusión de secuencias adicionales.

Filtrado de secuencias

Una vez obtenidos los sets de datos positivos y negativos a evaluar, se procedió a filtrar las secuencias disponibles, de acuerdo con los siguientes criterios:

1. Se decidió filtrar y seleccionar aquellas secuencias en los sets de datos positivo y negativo, cuyos rangos de tamaño estuvieran entre 50 a 150 nucleótidos. La mayoría de secuencias pre-miRNA anotadas en especies modelo se encontró situada en este rango de nucleótidos por secuencia, pese a que podría no haberse implementado tal rango de tamaño, o flexibilizarlo en función de tamaños más pequeños o más grandes. Para llevar a cabo este cometido, se ha desarrollado una función en lenguaje R, denominada *eMIRNA.Filter.by.Size.* Esta función adquiere cuatro variables *input* para su funcionamiento:

    - Dirección o *path* hacia el archivo FASTA con las secuencias positivas (pre-miRNAs) o negativas (no pre-miRNAs). Este archivo FASTA deberá poseer un formato no multilineal, con cada secuencia definida por un *header* o identificador (típicamente con el símbolo “>” como comienzo de línea, y una línea donde se defina la secuencia de nucleótidos a evaluar. En caso de que el archivo FASTA a utilizar, posea secuencias en formato multilineal, esto es, con secuencias divididas en sublíneas o secciones de 60 nucleótidos aproximadamente, podremos transformar y linealizar el archivo FASTA mediante el siguiente comando, que hace uso de los lenguajes Perl y awk de gestión y manipulación de datos:

      ```
      $ perl -pe ‘/^>/ ? print “\n” : chomp’ in.fasta | tail -n +2 | awk ‘{FS=” “}{print $1}’ > out.fa
      ```

      Donde `in.fasta` se corresponde a la dirección o *path* del archivo FASTA multilineal, y `out.fa` define el archivo de salida en formato FASTA lineal, preparado para ser filtrado por la función *eMIRNA.Filter.by.Size.*
    - Rango inferior de filtrado de secuencias por número de nucleótidos. En nuestro caso, se optó por establecerlo en 50 nucleótidos por secuencia.

- Rango superior de filtrado de secuencias por número de nucleótidos. En nuestro caso, se optó por establecerlo en 150 nucleótidos por secuencia.

- Prefijo o nombre identificador para el archivo FASTA, resultado del proceso de filtrado por tamaño de secuencia.

eMIRNA.Filter.by.Size, una vez definidas las variables necesarias para su funcionamiento, creará una carpeta en la *HOME/* del usuario, denominada *Sequence_FilterSize_Results/*, en la cual se guardará el archivo resultante del proceso de filtrado.

2. Por otra parte, y siguiendo la estrategia de filtrado por selección de atributos[94], se decidió implementar un segundo filtro de secuencias a partir de las ya seleccionadas por tamaño. Con el fin de homogeneizar lo más posible las secuencias negativas, y asimilar su estructura a la de las secuencias positivas anotadas como pre-miRNAs, se optó por seleccionar sólo aquellas secuencias negativas cuya estructura secundaria derivada del plegamiento de la cadena de ARN sobre sí misma, formara una estructura característica en horquilla, con dos brazos o *stems* unidos por una curva o *terminal loop*[113], asimilable a la adoptada por la inmensa mayoría de microRNAs anotados.

   Con este objetivo, se ha desarrollado la función *eMIRNA.Filter.by.Structure*. Esta función adquiere dos variables *input* para su funcionamiento:

   - Dirección o *path* hacia el archivo FASTA creado como resultado de aplicar la función *eMIRNA.Filter.by.Size*, anteriormente descrita.

   - Prefijo o nombre identificador para el archivo FASTA, resultado del proceso de filtrado por estructura de las secuencias.

   El resultado del proceso de filtrado mediante *eMIRNA.Filter.by.Structure*, es un nuevo archivo FASTA, en el cual se habrán eliminado todas aquellas secuencias cuyas predicciones de plegamiento en estructura secundaria contuvieran más de 1 *terminal loop,* conservándose sólo aquellas secuencias que posean una estructura en horquilla semejante a la característica forma de plegamiento que definen a los pre-miRNAs[113].

   Para la predicción de dicha estructura secundaria, *eMIRNA.Filter.by.Structure* hace uso del programa de plegamiento RNAfold[62], que realiza una estimación de la estructura de plegamiento con una mínima energía libre (*mínimum free energy*) y reporta una predicción estructural para dicha estructura. El programa RNAfold deberá ser guardado en una dirección o *path* incluida en el entorno principal o *$PATH* del sistema operativo. Todas aquellas secuencias que posean 1 terminal *loop*, serán seleccionadas para conformar un archivo

FASTA como resultado de aplicar eMIRNA.Filter.by.Structure.

eMIRNA.Filter.by.Structure, una vez definidas las variables necesarias para su funcionamiento, creará una carpeta en la *HOME/* del usuario, denominada *Sequence_FilterStructure_Results/*, en la cual se guardará el archivo resultante del proceso de filtrado.

2.2. Atributos estructurales

Una vez definidos los sets de datos positivos y negativos con las secuencias filtradas por tamaño y configuración estructural a evaluar, se plantea la necesidad de extraer una serie de atributos estructurales que definan y caractericen dichas secuencias, a fin de que puedan ser integrados en el proceso de entrenamiento de algoritmos basados en *Machine Learning*.

Cálculo de atributos

Siendo necesaria la representación cuantitativa de la estructura de las secuencias a analizar, con el objetivo de su posterior clasificación mediante el entrenamiento de algoritmos de *Machine Learning*, se decidió elaborar una función en el entorno de lenguaje R, a la que denominamos *eMIRNA.Features,* con el objetivo de reproducir el cálculo de atributos estructurales para nuestro trabajo.

eMIRNA.Features requiere de las siguientes variables para su funcionamiento:

- Dirección o *path* hacia el archivo FASTA creado como resultado de aplicar la función *eMIRNA.Filter.by.Structure*, anteriormente descrita.

- Nombre del archivo FASTA creado como resultado de aplicar la función *eMIRNA.Filter.by.Structure.*

- Prefijo o nombre identificador para el archivo output con extensión .csv, convertible a formato EXCEL, formado por la matriz de atributos calculados sobre cada una de las secuencias proporcionadas a la función.

- Pval = TRUE/FALSE. Por defecto, esta opción está desactivada (FALSE).

La función *eMIRNA.Features* se ha desarrollado para generar el cálculo de un total de 112 atributos estructurales para cada secuencia evaluada. De entre ellas, un total de 56 atributos de estructura de secuencia, que comprenden 7 variables diferenciadas, 23 atributos de estructura secundaria,

que comprenden un total de 8 variables, y un total de 33 atributos estadísticos de plegamiento de secuencia.

A continuación, se detallan los atributos estructurales calculados por *eMIRNA.Features:*

Atributos de estructura de secuencia:

- 32 elementos tripletes extraídos del cálculo del modelo SVM-Triplets[87] (T1 a T32).
- Longitud de la secuencia evaluada (Length).
- Ratio del contenido en Guanina-Citosina y longitud de secuencia (GC).
- Ratio del contenido en Guanina/Citosina (G.Cr)
- Ratio del contenido Adenina-Uracilo y Guanina-Citosina (AU.GCr).
- Ratios del contenido de cada base nitrogenada (A, U, G, C) y la longitud de secuencia (Ar, Ur, Gr, Cr).
- Ratios del contenido de dinucleótidos y longitud de secuencia (AAr, GGr, CCr, UUr, AGr, ACr, AUr, GAr, GCr, GUr, CAr, CGr, CUr, UAr, UGr, UCr).

Atributos de estructura secundaria:

- Longitud de la horquilla o *terminal loop* (Hl).
- Longitud de los brazos o *stems* 5' y 3' (Steml5, Steml3).
- Número de bases apareadas en la estructura secundaria (BP).
- Número de bases apareadas en los brazos o *stems* 5' y 3' (BP5, BP3).
- Número de bases no apareadas en los brazos o *stems* 5' y 3' (Mism5, Mism3).
- Número de burbujas o *bulges* en los brazos o *stems* 5' y 3' (Bulge5, Bulge3).
- Número de burbujas o *bulges* en los brazos o *stems* 5' y 3', de 1, 2, 3, 4 o 5 bases nitrogenadas (BulgeN1.5, BulgeN1.3 a BulgeN5.5, BulgeN5.3).
- Número de enlaces A-U, G-C y G-U (AUp, GCp, GUp).

Atributos estadísticos de plegamiento de secuencia:

- Energía libre mínima estimada por RNAfold[62] (MFE).
- Energía libre de ensamblado (EFE).
- Energía libre del Centroide (CFE).
- Distancia del Centroide al ensamblado (CDE).
- Energía libre de la estructura estimada con mayor fiabilidad (MEAFE).
- Máxima fiabilidad estimada (MEA).
- Predisposición de enlace entre bases (BPP).
- Frecuencia de la estructura MFE (EFreq).
- Diversidad estructural (ED).
- MFE ajustado por longitud de secuencia (MFEadj).
- EFE ajustado por longitud de secuencia (EFEadj).
- CDE ajustado por longitud de secuencia (Dadj).
- Entropía de Shannon ajustada por longitud de secuencia (SEadj).
- Diferencia entre MFE y EFE, ajustado por longitud de secuencia (DiffMFE.EFE).
- Ratio entre MFEadj y GC (MFEadj.GC).

- Ratio entre MFEadj y BP (MFEadj.BP).
- Energía libre mínima estimada por UNAFold[63] (dG).
- dG ajustado por longitud de secuencia (dGadj).
- Entropía estructural estimado por la función Melt de UNAFold (dS).
- dS ajustado por longitud de secuencia (dHadj).
- Entalpía estructural estimada por la función Melt de UNAFold (dH).
- dH ajustado por longitud de secuencia (dSadj).
- Temperatura de fusión estimada por la función Melt de UNAfold (dT).
- dT ajustado por longitud de secuencia (dTadj).
- MFE *P*-valor estimado mediante la generación de 100 secuencias aleatorizadas por el software uShuffle[114], y calculado según lo reportado por Jiang et al. 2007[79] (MFE.Pval).
- MFEadj Z-score (MFEz).
- MFEz *P*-valor (MFEz.Pval).
- EFE *P*-valor, calculado de forma análoga a MFE.Pval (EFE.Pval).
- EFEadj Z-score (EFEz).
- EFEz *P*-valor (EFEz.Pval).
- BPP Z-score (BPPz).
- Dadj Z-score (Dz).
- Dz *P*-valor (Dz.Pval).

Por otra parte, *eMIRNA.Features* requiere de los siguientes softwares externos y librerías de R para su funcionamiento:

Software:

- RNAfold[62], disponible en el siguiente enlace: https://www.tbi.univie.ac.at/RNA/

- UNAFold[63], disponible en el siguiente enlace: https://github.com/rcallahan/UNAFold

- Triplet-SVM perl scripts[87], disponibles en el siguiente enlace: https://github.com/PeterScott/mirna-prediction-nodejs/tree/master/progs/triplet-svm-classifier060304

- Ushuffle[114], disponible en el siguiente enlace: http://digital.cs.usu.edu/~mjiang/ushuffle

Librerías R:

- Sequinr (https://CRAN.R-project.org/package=sequinr)
- LncFinder (https://CRAN.R-project.org/package=LncFinder)
- Stringr (https://CRAN.R-project.org/package=stringr)
- Biobase(https://www.bioconductor.org/packages/release/bioc/html/Biobase.html)[115]
- Purrr (https://CRAN.R-project.org/package=purrr)
- Scales (https://CRAN.R-project.org/package=scales)
- Data.table (http://r-datatable.com)

Debido a la elevada intensidad de computación requerida para el cálculo de las iteraciones aleatorias sobre cada secuencia mediante el software uShuffle, sobre todo en el caso de evaluar más de 100 secuencias como input de la función *eMIRNA.Features*, se decidió deshabilitar el cálculo de las variables que hicieran uso de estas secuencias generadas a partir de la iteración aleatoria de las originales. No obstante, se dejó como variable opcional a activar en el caso de que se quisiera calcular todo el set disponible.

Una vez calculados todos los atributos estructurales, *eMIRNA.Features* normaliza todas las variables cuyo rango sea diferente [0, 1], aplicando la función *rescale(),* disponible en la librería *scales* de R, con el objetivo de normalizar la distribución de los atributos y evitar el efecto distorsionador de rangos dispersos en el ajuste del modelo.

Seguidamente al filtrado de los sets de datos positivos y negativos seleccionados, los archivos FASTA resultantes de aplicar la función *eMIRNA.Filter.by.Structure*, fueron introducidos en la función *eMIRNA.Features*, que generó las correspondientes matrices normalizadas para su uso posterior en el entrenamiento de algoritmos clasificadores de *Machine Learning*. Se optó por no incluir el cálculo de *P*-valores y variables derivadas del proceso de iteración sobre las secuencias, debido a la elevada intensidad computacional requerida, extrayéndose un total de 102 variables para cada uno de los sets de datos positivos y negativos evaluados.

2.3. Entrenamiento de algoritmos

Una vez generadas las matrices de atributos estructurales para cada uno de los sets de datos evaluados, se procedió a entrenar algoritmos clasificadores basados en *Machine Learning*, con el fin de obtener un modelo predictor capaz de clasificar secuencias pre-miRNA de forma eficaz.

Con este objetivo, se desarrolló la función en lenguaje R denominada *eMIRNA.Train.* Esta función adquiere como variables dos matrices de atributos estructurales extraídos a partir de secuencias positivas (pre-miRNAs) y negativas (no pre-miRNAs), resultado obtenido al ser evaluadas por la función *eMIRNA.Features*. Para el proceso de entrenamiento, *eMIRNA.Train* construye un set de datos con las matrices de atributos positiva y negativa, que es seguidamente subdividido en un set de datos de entrenamiento y un set de datos de testeo. Esta subdivisión se realiza de forma aleatoria, tal que el set de datos de entrenamiento suponga un 80% del total de datos positivos y negativos calculados, mediante el uso de la función *createDataPartition* del paquete de R *caret*[116]*.* Seguidamente procede al entrenamiento del modelo de *Support Vector Machine* (SVM), haciendo uso así mismo de la función *svmLinear* incluida en el paquete *caret,* y aplicando un modelo no exhaustivo de *cross-validation,* conocido como *k-fold cross-validation.*

Algoritmo Support Vector Machine

El algoritmo SVM[117] puede ser entendido como un método de aprendizaje supervisado adecuado para procesos de clasificación y regresión. En nuestro caso, lo hemos aplicado al problema de clasificación de secuencias estructuralmente compatibles con pre-miRNAs o no. Dados una serie de datos agrupados en categorías, el modelo algoritmo SVM construirá un modelo de asignación para nuevos elementos a una categoría u otra. Para ello, el modelo utiliza la representación de los datos aportados como puntos en un plano multidimensional que determinan los valores de sus atributos estructurales. Cada dato o punto dimensional representativo es considerado como un vector p-dimensional (una lista de p números). El objetivo del modelo será intentar separar el conjunto de puntos en un hiperplano dimensional, creando una división homogénea de datos en cada lado del hiperplano.

En el caso de un modelo de clasificación binaria, el algoritmo tratará de identificar la línea del hiperplano que divida ambas clases. Para ello, el modelo definirá el hiperplano de tal forma que la distancia de este a cada punto o dato más cercano para cada clase, sea máxima, concepto definido como la búsqueda del *Maximum Margin Hyperplane* (MMH), que determinará, si existe, el *Maximum Margin Classifier.* Por otra parte, los vectores de soporte del hiperplano, o *support vectors,* se identificarán como los puntos o datos de cada clase más cercanos al MMH, siendo necesaria la presencia de al menos un vector de soporte para cada una de ellas, aunque pueden definirse más de uno. Mediante los vectores de soporte, podremos definir espacialmente el MMH.

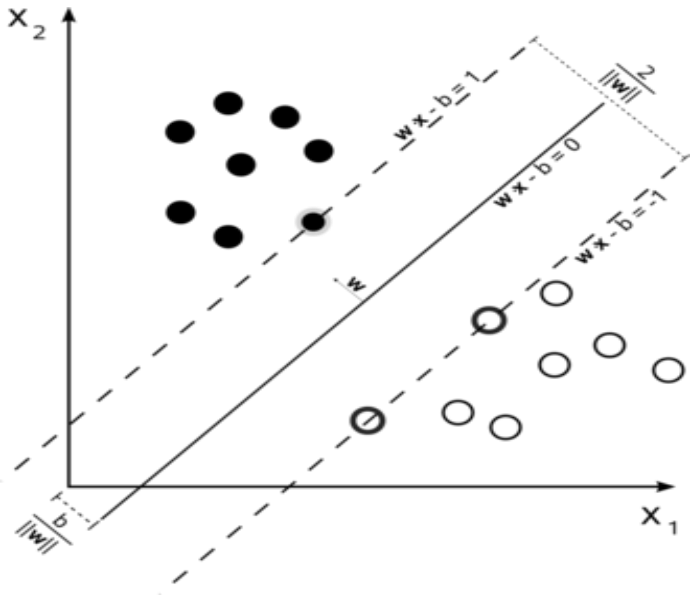

Figura 7: Representación gráfica del MMH y márgenes en un modelo SVM con dos clases. Los puntos situados en los márgenes representan los *support vectors.*

Asumiendo la posibilidad de que ambas clases o categorías a clasificar puedan ser separadas linealmente, el MMH se definirá en un espacio tan alejado de los márgenes exteriores de cada grupo como sea posible. Estos márgenes exteriores son conocidos como el *convex hull*, y el MMH determinará el bisector perpendicular a la línea más corta entre ambos *convex hulls*. Otra posibilidad incluye la búsqueda espacial de cada uno de los hiperplanos posibles con el objetivo de encontrar dos planos paralelos capaces de dividir los puntos en el espacio en dos clases homogéneas, estando tan alejados como sea posible.

Definiendo el hiperplano como:

$$\vec{w} \cdot \vec{x} + b = 0$$

Donde *ω* define un vector con *n* variables, y *b* define el sesgo o *bias*, el objetivo del proceso de búsqueda del MMH será la búsqueda de una serie de variables o datos que definan dos hiperplanos tal que:

$$\vec{w} \cdot \vec{x} + b \geq +1$$
$$\vec{w} \cdot \vec{x} + b \leq -1$$

De este modo, se definirán los hiperplanos bajo la premisa de que todos los puntos de una clase se concentren por encima del primer hiperplano, y que los pertenecientes a la otra clase, se concentren por debajo del segundo hiperplano, en tanto que el conjunto de datos sea linealmente separable. El objetivo último será minimizar la distancia entre los dos hiperplanos, definida como:

$$\frac{2}{||\vec{w}||}$$

Donde el vector ||ω|| determina la *Euclidean norm* (distancia desde el origen al vector ω. Esta minimización puede ser expresada tal que:

$$\min \frac{1}{2} ||\vec{w}||^2$$
$$s.t. \;\; y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1, \forall \vec{x}_i$$

Donde la primera expresión determinada la minimización del *Euclidean norm*, en tanto que la segunda expresión condiciona esta minimización a que cada uno de los y_i puntos sea correctamente clasificado.

k-fold Cross-Validation

Durante el proceso de entrenamiento del modelo, aplicaremos el proceso de validación conocido como *k-fold cross-validation*, donde el conjunto de secuencias utilizadas como set de entrenamiento, es particionado en *k*

submuestras de igual tamaño. Para cada una de las k submuestras, una de ellas será utilizada como el conjunto de datos de validación para el testeo del modelo, mientras que el resto, *k* – 1 submuestras, serán utilizadas como datos de entrenamiento. Posteriormente el proceso de *cross-validation* será repetido *k* veces, seleccionando cada una de las *k* submuestras como conjunto de datos de testeo, una vez. Finalmente, los *k* resultados generados para el proceso de clasificación serán ponderados para producir una única estimación. Una de las principales ventajas de este método radica en que todas y cada una de las observaciones o secuencias serán utilizadas tanto para el proceso de entrenamiento como para el de validación, siendo seleccionada cada una de las observaciones exactamente una vez para el proceso de validación.

eMIRNA.Train aplica un proceso de *10-fold-crossvalidation*, considerado uno de los valores *k* más utilizados[118]. Una vez entrenado el modelo, podremos evaluar el nivel de eficacia alcanzado durante el proceso de clasificación.

Por otra parte, se analizó la relevancia de cada uno de los atributos estructurales calculados, en el proceso de clasificación, mediante el entrenamiento de un modelo de predicción basado en el algoritmo de *Machine Learning Random Forest.*

El algoritmo *Random Forest* (RF) fue originalmente propuesto por Breiman L. 2001 (Breiman L. 2001. Random Forest), fundamentado en la combinación de clasificadores estructurados en forma de árbol, conjuntamente con la aleatoriedad y robustez proporcionada por los sistemas de *bagging* y selección aleatoria de atributos. Durante el proceso de entrenamiento, multitud de árboles de decisión son entrenados con muestreos aleatorios por Bootstrap a partir del set de datos original, para posteriormente combinar los resultados en una única predicción. En particular, este tipo de algoritmo no utiliza todos los atributos para desarrollar los árboles de decisión, sino que selecciona una muestra aleatoria de atributos para cada nodo de decisión, eligiendo aquella que genere una mejor partición de los datos, para generar el siguiente nodo de decisión.

2.4. Detección por homología

Una vez entrenado el modelo SVM de clasificación, y analizados los atributos que mostraron mayor relevancia en el proceso de separación de clases mediante el algoritmo Random Forest (RF), se procedió a identificar y seleccionar posibles secuencias a evaluar y clasificar, con el objetivo de detectar nuevas estructuras pre-miRNAs no anotadas en la especie porcina.

Con este objetivo, se decidió hacer uso de la mayor y más extensa y consolidada anotación de microRNAs en la especie humana, para realizar búsqueda pormenorizada y comparativa de miRNAs anotados en el genoma porcino, tomando como base la detección de estructuras por homología, gracias a la sólida conservación filogenética y escasa variación evolutiva para

las formas maduras de los miRNAs, y en especial para las regiones *seed* de las mismas, determinantes de su función biológica. Para esta tarea se ha desarrollado un programa de alineamiento, detección y filtrado de secuencias, denominado *eMIRNA-Hunter.* Este programa hace uso de los softwares Bowtie[119] y BEDTools v2.27.0[120] para su funcionamiento, que deberán ser instalados en una dirección o *path* incluida en el entorno principal o *$PATH* del sistema operativo. Será necesario construir los correspondientes índices para Bowtie a partir del genoma de referencia a utilizar, en nuestro caso, el *assembly* porcino en su versión 11.1.

eMIRNA-Hunter requiere los siguientes archivos *input*:

- Archivo FASTA con secuencias de microRNAs maduros en humano, en código ADN (U > T). En la base de datos miRBase, podremos encontrar un archivo FASTA con todas las secuencias miRNAs maduras anotadas, a partir del cual se han filtrado aquellas pertenecientes a *Homo sapiens* (prefijo hsa-):
  http://www.mirbase.org/ftp.shtml

- *Path* o dirección donde se haya generado el *index* para Bowtie a partir del genoma de referencia a utilizar en el proceso de alineamiento.

- *Path* o dirección donde guardar los archivos generados como *output*.

- Prefijo o nombre identificativo para los archivos generados como *output*.

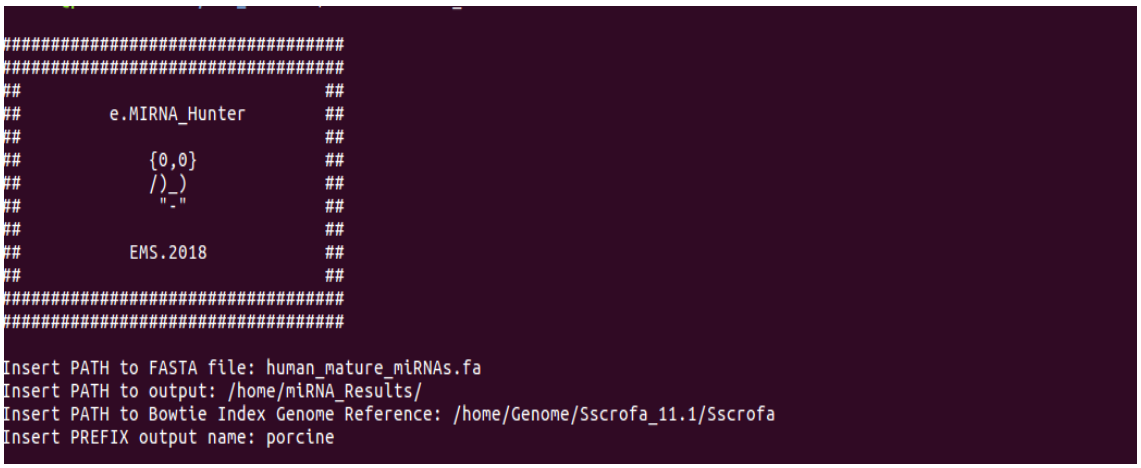

Figura 8: Ejemplo de ejecución del programa *eMIRNA-Hunter*.

La metodología de *eMIRNA-Hunter* parte de la realización de un alineamiento a partir de las secuencias maduras de miRNAs para Homo sapiens, aportadas como input, frente al genoma de referencia elegido para el estudio, que, en nuestro caso, fue el *assembly* porcino en su versión 11.1. El proceso de alineamiento se lleva a cabo mediante el uso del software Bowtie.

Este programa realiza un alineamiento optimizado para grandes cantidades de secuencias cortas de ADN respecto a un genoma de referencia de nuestra elección. Puede alinear secuencias de en torno a 35 nucleótidos a un ritmo de 25 millones de secuencias por hora en un equipo promedio. Bowtie necesita generar un índice del genoma de referencia mediante el algoritmo de *Burrows-Wheeler*. *eMIRNA-Hunter* implementa un alineamiento optimizado para microRNAs[121,122] (bowtie -f -p 16 -n 1 -l 10 -m 20 -k 1 –best –strata), y genera un archivo en formato SAM con las secuencias de miRNAs maduros alineados respecto al genoma de referencia elegido. A continuación, se realiza una selección de aquellas secuencias que alinearon de forma exitosa, y se hace uso de las posiciones de alineamiento identificadas para la construcción de un archivo de posiciones estimadas de alineamiento en el genoma de referencia. Estas posiciones genómicas son posteriormente elongadas en sentido 5' y 3', con el objetivo de reconstruir potenciales posiciones para secuencias pre-miRNAs, que serán extraídas a partir del genoma de referencia mediante el uso del software bedtools, hacia un archivo de posiciones en formato BED, y así mismo guardadas como tales secuencias en un archivo con formato FASTA, que podremos utilizar a continuación como referencia para su clasificación mediante el algoritmo SVM anteriormente entrenado.

Como resultados, *eMIRNA-Hunter* proporcionará 4 archivos *output*:

- Archivos en formato BED y FASTA con las secuencias alineadas y elongadas para reconstruir potenciales estructuras pre-miRNAs.
- Archivo SAM generado a partir del alineamiento con el software Bowtie.
- Archivo LOG de estadísticas y proceso del alineamiento.

2.5. Clasificación de secuencias

Una vez extraídas las potenciales estructuras pre-miRNAs a partir del alineamiento y filtrado de secuencias homólogas, se podrá hacer uso del archivo FASTA generado por *eMIRNA-Hunter*, que contiene todas las secuencias de microRNAs humanos elongadas que pudieron alinear en el *assembly* porcino, para ser clasificadas por el modelo SVM entrenado previamente mediante la función *eMIRNA.Train*. Para llevar a cabo esta tarea, se ha definido una función en lenguaje R, denominada *eMIRNA.Predict.* Haciendo uso de la función de R *predict*, elabora una clasificación del set de datos introducidos a partir del modelo SVM de *Machine Learning* entrenado.

Una vez obtenidas las secuencias a clasificar, se filtrarán por tamaño y estructura haciendo uso de las funciones *eMIRNA.Filter.by.Size* y *eMIRNA.Filter.by.Structure*, descritas anteriormente, y se generarán las matrices de atributos estructurales a partir de la función eMIRNA.Features. La matriz resultante de atributos estructurales seguidamente será utilizada como

input para la clasificación de las secuencias evaluadas.

eMIRNA.Predict requiere las siguientes variables para su funcionamiento:

- Objeto R conteniendo el modelo SVM entrenado por *eMIRNA.Train*.
- Matriz de atributos estructurales calculada a partir de las secuencias extraídas por *eMIRNA-Hunter*.
- Prefijo o nombre identificativo para los archivos generados como *output*.

Como resultado, *eMIRNA.Predict* generará una carpeta denominada *Prediction_Results/,* alojada en la *HOME/* del usuario, donde se guardará un archivo de texto con el listado de identificadores de estructuras pre-miRNAs humanos y homólogos en porcino, que fueron clasificados como tales por el algoritmo clasificador SVM entrenado.

2.6. Anotación de nuevos miRNAs

Seguidamente a la clasificación de secuencias como posibles candidatos a pre-miRNAs, se hace necesario discriminar de entre todas estas secuencias, aquellas ya anotadas en porcino, de aquellas otras que podrían suponer nuevos candidatos a microRNAs en el genoma porcino aún no anotados. Cabe destacar que mediante el proceso de extracción de secuencias por *eMIRNA-Hunter*, habremos seleccionado todas aquellas secuencias homólogas a microRNAs anotados en humano, que fueron capaces de alinear en el assembly porcino, sin discriminar entre secuencias ya anotadas en porcino o no.

Una vez obtenida la lista de microRNAs predichos por *eMIRNA.Predict*, se hace necesario discriminar cuáles de ellos ya han sido anotados de aquellos que aún no lo han sido, y, de entre los últimos, cuáles poseen una mayor probabilidad de suponer una nueva secuencia compatible con la estructura de un microRNA, aún no anotada en el genoma porcino.

Con este objetivo, se ha desarrollado el programa *eMIRNA-Seeker.* Este programa realiza una primera clasificación entre microRNAs predichos anotados, y no anotados. A continuación, y centrándose en aquellos aún no anotados, identifica las regiones genómicas donde han sido identificados, conjuntamente con las regiones de sus correspondientes homólogos en la anotación de humano, estableciendo una ventaja de búsqueda de elementos codificantes de 2 Megabases (Mb) en sentido 5’ y 3’ respecto a las posiciones de cada pre-miRNA no anotado. Esta ventana de posición es seguidamente utilizada por un script R auxiliar denominado *biomaRt_calc.R*, que realiza una búsqueda mediante el paquete de R biomaRt[123] sobre cada una de las ventanas fijadas en el entorno de 2Mb para la posición en el genoma de cada

pre-miRNA no anotado en porcino, y para su correspondiente homólogo anotado en humano, extrayendo los elementos codificantes presentes dentro de estas ventanas genómicas.

Por último, *eMIRNA-Seeker* realiza un cálculo de probabilidad de vecindad genómica (*Neighbouring Score*), fundamentado en la asunción de que para un microRNA dado no anotado en porcino, la probabilidad de que éste sea considerado como un fiel candidato a microRNA y no una secuencia aleatoria extraída y predicha por azar, es tanto mayor cuanto más coincidente sea su vecindad génica con respecto a la anotación en la especie modelo de referencia, en este caso, el humano. Con este principio, *eMIRNA-Seeker* filtra todos aquellos miRNAs candidatos no anotados para los que no se haya detectado ninguna coincidencia en el entorno genómico entre humano y la especie analizada, porcino en nuestro caso, centrándose en los miRNAs candidatos que al menos muestran 1 coincidencia génica homóloga en la ventana de 4 Mb cuyo centro es el propio miRNA candidato. Finalmente, a partir de los miRNAs seleccionados, calcula un valor probabilístico o *Neighbouring Score*, como la razón entre el número de coincidencias entre genes homólogos de humano y porcino, y el número de genes anotados en la ventana de 4Mb establecida en el genoma porcino para cada miRNA candidato.

Este valor de probabilidad o *Neighbouring Score* tendrá un rango de 0 a 1, representando el 0 la menor probabilidad de considerar al miRNA como un candidato fiable, y, al contrario, el valor 1 como la mayor probabilidad de que el miRNA detectado sea un firme candidato a una nueva anotación en el *assembly* porcino.

eMIRNA-Seeker hace uso de los softwares BEDTools v2.27.0[120] y BEDOPS[124]. Por otra parte, requiere los siguientes archivos *input:*

- Archivo de anotación génica en formato GTF para el *assembly* de la especie en estudio, el porcino en su versión 11.1 en nuestro caso, disponible en: ftp://ftp.ensembl.org/pub/release-92/gtf/sus_scrofa/

- Archivo de anotación de microRNAs en la especie modelo de referencia, Homo sapiens, disponible en: ftp://mirbase.org/pub/mirbase/CURRENT/genomes/

- Listado de estructuras pre-miRNA, generado como output por *eMIRNA-Hunter.*

- Archivo en formato BED con posiciones para microRNAs homólogos a humano en porcino, generado como output por *eMIRNA-Hunter.*

- *Path* o dirección donde guardar los archivos generados como *output*.

- Prefijo o nombre identificativo para los archivos generados como *output*.

Como resultado de aplicar el filtrado y cálculo de probabilidad y fiabilidad de los candidatos a microRNAs no anotados en porcino, *eMIRNA-Seeker* genera los siguientes archivos como *output:*

- Archivo en formato BED, con las posiciones asociadas a los pre-miRNAs homólogos de humano detectados por *eMIRNA.Predict*.

- Archivo en formato BED con las posiciones asociadas a los pre-miRNAs homólogos de humano detectados por *eMIRNA.Predict*, y que ya aparecen como estructuras miRNA anotadas en el *assembly* porcino 11.1.

- Archivo en formato BED con las posiciones asociadas a los pre-miRNAs homólogos de humano detectados por *eMIRNA.Predict*, y que no se corresponden con miRNAs anotados en el *assembly* porcino 11.1.

- Archivo resultante del filtrado de aquellos candidatos probables a nuevos microRNAs en el genoma porcino, detallando su posición (cromosoma, inicio y final de la secuencia pre-miRNA), nombre del candidato a miRNA y valor de fiabilidad o *Neighbouring Score.*

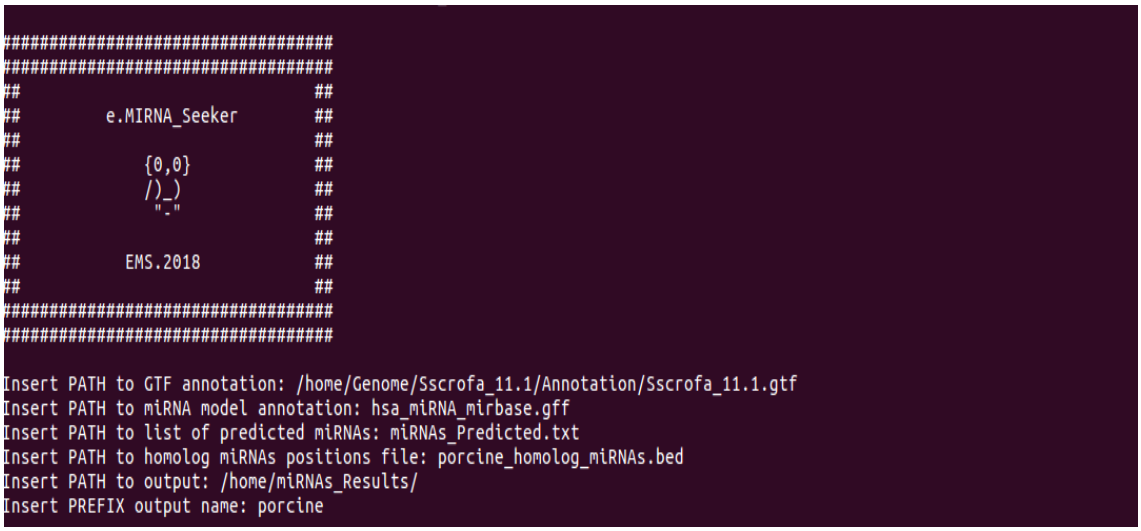

Figura 9: Ejemplo de ejecución del programa *eMIRNA-Seeker*.

3. Resultados y Discusión

Datos positivos y negativos

Una vez aplicados los filtros por tamaño de secuencia y estructura de plegamiento descritos en el desarrollo de las funciones *eMIRNA.Filter.by.Size* y *eMIRNA.Filter.by.Structure*, a los sets de datos positivos y negativos evaluados, e incluyendo los datos de estructuras mock-miRNAs generados para el set de datos de porcino, se obtuvieron los siguientes resultados:

- Set de datos positivos de secuencias pre-miRNAs, anotadas en Ensembl para el *assembly* porcino 11.1, con un total de 460 secuencias.
- Set de datos negativos de secuencias no pre-miRNAs, anotadas en Ensembl para el *assembly* porcino 11.1, con un total de 361 secuencias.
- Set de datos positivos de secuencias pre-miRNAs, anotadas en Ensembl para el *assembly* humano h38.p12, con un total de 1812 secuencias.
- Set de datos negativos de secuencias no pre-miRNAs, anotadas en Ensembl para el assembly humano h38.p12, con un total de 2587 secuencias.
- Set de datos negativos de pseudo-miRNAs filtradas a partir de los datos publicados por Ng *et al*. 2007[98], con un total de 7185 secuencias.

Una vez definidos los sets de secuencias positivas y negativas a evaluar, se procedió al cálculo de los atributos estructurales mediante la función *eMIRNA.Features*. Para el caso del set de datos negativos de pseudo-miRNAs, debido al elevado número de secuencias resultantes en comparación a los sets de datos para porcino y humano, y con el objetivo de evitar fenómenos de *class-imbalance*, se procedió a extraer sendas muestras de 460 y 2588 secuencias respectivamente, mediante muestreo aleatorio simple, sin reemplazo, evitando la posible repetición de secuencias, a partir de las cuales se procedió al cálculo de atributos estructurales.

A continuación, se procedió a entrenar un modelo SVM lineal mediante el uso de las matrices de atributos calculados, utilizando la función *eMIRNA.Train.* Este proceso se realizó con las siguientes combinaciones de sets de datos:

- Sets de datos positivos y negativos para el *assembly* porcino 11.1, con 460 y 361 secuencias respectivamente.

- Sets de datos positivos y negativos para el *assembly* humano h38.p12, con 1812 y 2587 secuencias respectivamente.

- Set de datos positivos para el *assembly* porcino 11.1, y 460 secuencias aleatoriamente seleccionadas a partir del set de datos negativos pseudo-

miRNAs

- Set de datos positivos para el *assembly* humano h32.p12, y 2587 secuencias aleatoriamente seleccionadas a partir del set de datos negativos pseudo-miRNAs.

Evaluación de modelos

Se evaluaron los modelos SVM entrenados sobre los diferentes sets de datos positivos y negativos generados, obteniéndose la máxima precisión o *Accuracy* (100%) durante el proceso de clasificación, con un valor *kappa* = 1, para los cuatro contrastes propuestos (**Tabla 2**), confirmándose la robustez obtenida en el proceso de entrenamiento de los modelos SVM lineales mediante el uso de la función *eMIRNA.Train*.

Es importante resaltar que la elevada robustez en el poder de predicción y separación de clases mostrada por ambos modelos entrenados, supuso un resultado tomado con precaución en un principio. Una vez revisados el proceso de partición de datos para el entrenamiento, así como el método de muestreo por cross-validation, se obtuvieron los mismos resultados. Este elevado poder de clasificación ya ha sido reportado en otros trabajos previos[74], aunque no con tan elevados porcentajes de precisión.

Una de las posibles explicaciones a estos datos estriba en el proceso de selección de datos positivos y negativos para el entrenamiento de los algoritmos SVM y RF, donde pese a haberse realizado un filtrado por tamaño de secuencia y estructura de plegamiento, la diferencia entre clases positiva y negativa puede aún resultar bastante marcada, facilitando así la clasificación, conjuntamente con la gran variedad de atributos estructurales y estadísticos extraídos para cada secuencia.

Model - Constrast	Accuracy	*kappa*
SVM – Pig positive & negative	1	1
RF – Pig positive & negative	0.9985	0.9969
SVM – Pig positive & pseudo-miRNAs	1	1
RF – Pig positive & pseudo-miRNAs	0.9878	0.9756
SVM – Human positive & negative	1	1
RF – Human positive & negative	0.9787	0.9559
SVM – Human positive & pseudo-miRNAs	1	1
RF – Human positive & pseudo-miRNAs	0.9974	0.9947

Tabla 2: Resultados de *Accuracy* y valor *kappa* para el entrenamiento de los modelos SVM lineal y *Random Forest.*

Con el objetivo de analizar los atributos más importantes o determinantes durante el proceso de clasificación, se utilizó el modelo de *Machine Learning Random Forest*, con 1500 árboles de decisión.

De los resultados obtenidos (**Figura 10**), se pudo comprobar que las variables o atributos estructurales MFEadj.GC y BPP resultaron ser consistentemente aquellas seleccionadas como las más relevantes durante el proceso de clasificación mediante el algoritmo RandomForest. No obstante, dada la variabilidad de la posición relativa en cuanto a importancia asociada para cada uno de los contrastes de datos propuestos, se podría deducir que la distribución y características intrínsecas de los datos evaluados tendría una gran influencia sobre el proceso de clasificación y la importancia relativa otorgada a los diferentes atributos estructurales calculados, destacándose sobre todo en el caso de los datos negativos. Resulta particularmente interesante que el proceso de clasificación con los datos positivos y negativos procedentes del genoma porcino, así como la clasificación con datos positivos para humano y porcino, y negativos para pseudo-miRNAs, otorgue una distribución de importancia de atributos semejante, siendo el análisis con datos positivos y negativos para el genoma humano, el que aportó una distribución de importancia de atributos, divergente del resto, pese a que las variables MFEadj.GC y BPP se ordenaron en vigésimo y undécimo lugar en importancia, respectivamente.

De estas observaciones debería extraerse cierta precaución en cuanto a priorizar el cálculo de ciertas variables o atributos sobre otras, siendo particularmente interesante ampliar el espectro de atributos estructurales extraídos, y tomar especial atención a la generación y evaluación de datos negativos, resultados en concordancia con otros trabajos anteriores donde se evaluó el poder discriminante de atributos estructurales para la clasificación e identificación de pre-miRNAs[125].

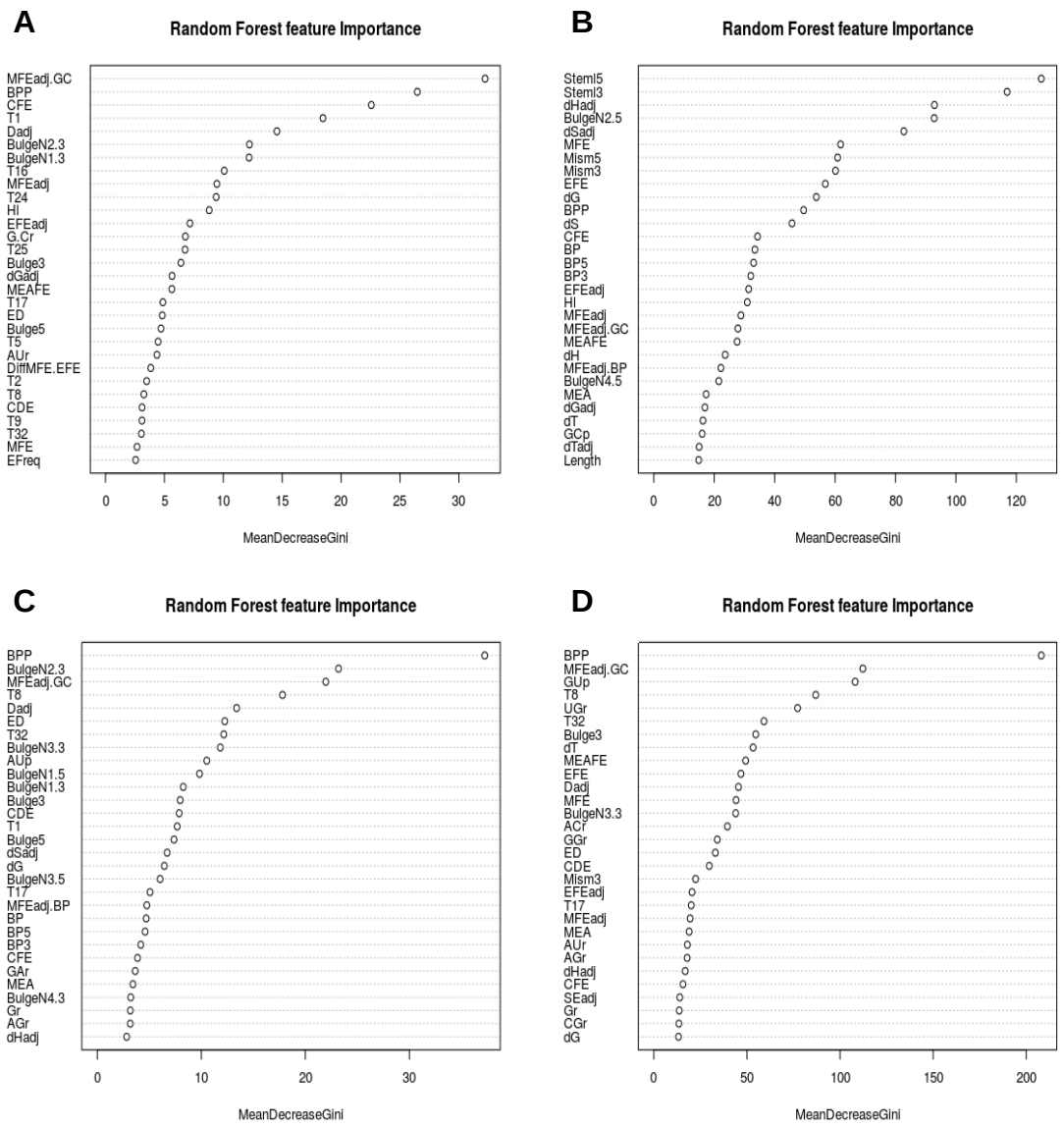

Figura 10: *Importance Plots* a partir de los modelos *Random Forests* entrenados. **A:** RF – Pig Positive & negative. **B:** RF – Human Positive & negative. **C:** RF – Pig positive & pseudo-miRNAs. **D:** RF – Human positive & pseudo-miRNAs.

Detección de miRNAs por homología

A partir de las secuencias anotadas para microRNAs maduros en humano, sumando un total de 2718 miRNAs diferentes, el proceso de alineamiento respecto al *assembly* porcino 11.1 aplicado por el script *eMIRNA-Hunter* resultó en 1774 secuencias correctamente alineadas, para las cuales se encontró una suficiente homología de secuencia, lo que supuso una tasa de éxito de alineamiento del 66,79%, y generándose un set de secuencias candidatas elongadas de 2228 pre-miRNAs. Podrían plantearse alineamientos menos restrictivos al utilizado por *eMIRNA-Hunter*, con el objetivo de incrementar el número de secuencias alineadas, no obstante, hemos

considerado aceptable el porcentaje de alineamiento conseguido, teniendo en cuenta el hecho de haber alineado secuencias correspondientes a humano, respecto al *assembly* porcino 11.1.

Estas secuencias candidatas fueron sometidas a filtrado y extracción de atributos estructurales, a partir de las funciones *eMIRNA.Filter.by.Size, eMIRNA.Filter.by.Structure* y *eMIRNA.Features*, resultando en un total de 1075 secuencias evaluadas (48,25% de secuencias superaron los filtros por longitud de secuencia y estructura), tras lo cual, la matriz de atributos resultantes fue introducida en el modelo de predicción entrenado por *eMIRNA.Train* para el set de datos positivos y negativos en el *assembly* porcino.

El proceso de clasificación arrojó un total de 383 secuencias identificadas como posibles pre-miRNAs, lo que supuso una tasa de éxito en la detección de estructuras pre-miRNAs del 21,59% sobre el total de secuencias alineadas con éxito.

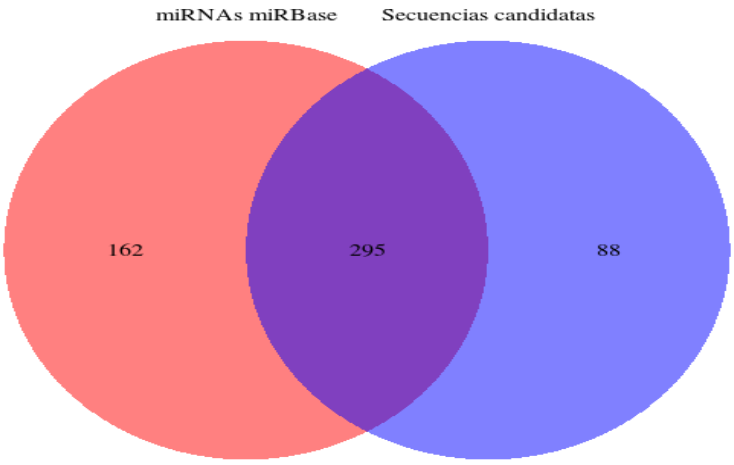

Figura 11: Diagrama de Venn mostrando el total de secuencias miRNA maduras anotadas en miRBase (rojo), respecto a las secuencias candidatas detectadas (azul). Un total de 88 correspondieron a secuencias no anotadas previamente.

Una vez clasificadas las secuencias pre-miRNAs extraídas a partir del contraste por homología respecto al genoma humano, se procedió a aplicar el script *eMIRNA-Seeker*, con el objetivo de diferenciar aquellas correspondientes a secuencias ya anotadas en el *assembly* porcino, de aquellas aún no anotadas y por lo tanto candidatas a nuevas estructuras pre-miRNAs. Del total de 383 secuencias clasificadas como pre-miRNAs por el modelo SVM de *Machine Learning*, 295 correspondieron a secuencias clasificadas como ya anotadas en el *assembly* porcino 11.1, mientras que un total de 88 se identificaron con regiones no anotadas, lo que supuso una tasa del 22,98% de detección de nuevos candidatos a microRNAs sobre el total de secuencias clasificadas como pre-miRNAs por el algoritmo SVM.

Por último, del proceso de cálculo de los valores de verosimilitud o fiabilidad respecto a la vecindad génica de los candidatos a microRNAs no anotados (*Neighbouring Score*), se seleccionaron un total de 26 secuencias con una elevada probabilidad de corresponder a estructuras homólogas de microRNAs humanos no anotados en el *assembly* porcino 11.1. No obstante, un análisis pormenorizado identificó algunas de ellas como secuencias previamente anotadas en el *assembly* porcino 10.2 y en la base de datos miRBase[58], pero no así en el reciente *assembly* 11.1. Por otra parte, otras secuencias se correspondieron con regiones anotadas para microRNAs con otra nomenclatura (**Tabla 3**).

Chr	Start	End	Strand	miRNA	N. Score	Observaciones
1	191218565	191218649	+	ssc-miR-3529	0.300	
2	1473423	1473504	-	ssc-miR-483	0.853	
2	65308297	65308378	+	ssc-miR-27a	0.917	anotado en miRBase
3	42018968	42019052	-	ssc-miR-3651	0.189	
5	95548379	95548461	+	ssc-miR-3059	1.000	
6	56426934	56427015	-	ssc-miR-520e	0.244	
6	63490750	63490832	+	ssc-miR-200a	0.549	
7	24044458	24044540	-	ssc-miR-1236	0.813	
7	34521753	34521835	+	ssc-miR-9983	0.733	
8	1205686	1205767	-	ssc-miR-4800	0.852	
8	30192280	30192362	+	ssc-miR-574	0.923	anotado en miRBase
9	114528004	114528085	+	ssc-miR-3120	0.633	ssc-miR-214
10	27079411	27079493	-	ssc-miR-24-1	0.765	ssc-miR-24-2
12	1601434	1601517	-	ssc-miR-3065	0.853	ssc-miR-338
12	45597380	45597461	+	ssc-miR-4523	0.762	
12	46211517	46211601	-	ssc-miR-3184	0.563	ssc-miR-423
12	48162621	48162703	-	ssc-miR-132	0.813	anotado en miRBase
13	33152294	33152378	+	ssc-miR-4787	0.786	
13	197168809	197168891	+	ssc-miR-6501	0.966	
14	109233955	109234040	-	ssc-miR-3085	0.952	
14	122706278	122706359	-	ssc-miR-6715b	0.960	
14	122706280	122706361	+	ssc-miR-6715a	0.960	
14	127016706	127016788	-	ssc-miR-9851	0.833	
16	38580342	38580424	+	ssc-miR-582	0.818	ssc-miR-582-1
17	58970317	58970399	-	ssc-miR-296	0.800	anotado en mirBase
X	94122536	94122619	+	ssc-miR-1264	0.900	

Tabla 3: Listado de pre-miRNA predichos mediante homología por el modelo SVM, que mostraron un *Neighbouring Score* (N. Score) confiable para ser considerados como candidatos a nuevos microRNAs en el genoma porcino.

De entre todas las secuencias seleccionadas, hemos considerado para su posterior análisis y como candidato de especial relevancia, el miRNA ssc-miR-483, homólogo del miRNA homónimo hsa-miR-483.

Adicionalmente, y para contrastar la fiabilidad de la secuencia de ssc-miR-483 como candidato pre-miRNA, se utilizaron dos algoritmos de predicción de estructuras pre-miRNA publicados anteriormente, RNAmicro[67], basado en la detección de microRNAs mediante homologías y el entrenamiento de un modelo de *Machine Learning* SVM, y miRClassify[80], que utiliza un modelo *Random Forest* para calcular la verosimilitud de las secuencias analizadas. Ambos algoritmos predijeron la secuencia del candidato ssc-miR-483 como compatible con una estructura pre-miRNA.

Ssc-miR-483 en el genoma porcino

El microRNA 483 destaca por ser un miRNA de origen intrónico, alojado en el intrón 2 del gen *Insuline Growth Factor 2* (IGF2), y ha sido descrito en diversas especies modelo y de interés comercial como humano (hsa-miR-483), ratón (mmu-miR-483), vaca (bta-miR-483), caballo (eca-miR-483) o cabra (chir-miR-483).

El algoritmo de predicción desarrollado pudo identificar el pre-miRNA candidato ssc-miR-483 en el assembly porcino 11.1, alojado en el cromosoma 2, en el interior del intrón 2 del gen IGF2 porcino (ENSSSCG00000035293), concretamente en posición 2:1473423-1473504(-), análoga a la descrita para hsa-miR-483 en humano y otras especies domésticas de interés. Por otra parte, pudimos refinar la anotación de los extremos del pre-miRNA candidato ajustando su posición a 2:1473425-1473498(-), evitando así los extremos no apareados detectados.

>ssc-miR-483 2:1473423-1473504(-)
GGAGGCGAGGGCGAGGACGGGAAGAGAGGA
GGGCGTGGTTTCTGCTGGTCCTCACTCCTCTC
CTCCCGTCTTCCTCCTCCT

Figura 12: Detalle de la secuencia y posición detectada para el candidato ssc-miR-483.

Figura 13: Representación estructural del candidato ssc-miR-483, realizada mediante el software RNAfold[62], donde se puede apreciar la configuración característica perteneciente a una secuencia pre-miRNA, con dos *stems* o brazos y un *terminal loop* en forma de horquilla.

Funcionalmente, miR-483 ha sido relacionado con procesos de regulación del crecimiento celular, e íntimamente asociado a mecanismos de proliferación tumoral, actuando como inhibidor de señales apoptóticas mediante la interacción con proteínas proapoptóticas como BBC3/PUMA[126] o DPC4/Smad4[127], así como inhibiendo la expresión de elementos antitumorales como EI24[128], considerándose como un elemento regulador con actividad oncogénica en la evolución de diversos procesos tumorales como el carcinoma esofágico de células escamosas[128], tumores pancreáticos[129], carcinomas adrenocorticales[130] o adenocarcinomas pulmonares[131]. Por otra parte, también se ha descrito un efecto contrario, actuando como inhibidor de la proliferación celular y como marcador y posible diana terapéutica en el tratamiento de cáncer de colon[132].

Su expresión ha sido asociada íntimamente a la expresión del gen IGF2[133], facilitando mediante la unión a factores de transcripción, la propia expresión de IGF2, en un ciclo de *feed-back* positivo. Particularmente interesante, además, resulta su posible asociación con procesos de resistencia a la insulina y susceptibilidad al síndrome metabólico. Fernand-McCollough et al. 2012[134], detectaron un incremento de la expresión de la porción 3' de miR-483 (miR-483-3p) en pacientes y ratones sometidos a carencias nutricionales en periodos tempranos del desarrollo, que ha sido relacionado con una mayor susceptibilidad a padecer patologías de tipo coronario o una mayor predisposición a desarrollar diabetes de tipo II. El aumento de la expresión de miR-483-3p podría desencadenar efectos represores indeseados sobre factores de transcripción o diferenciación celular como GDF3, que podría actuar como diana de miR-483-3p, dificultando o inhibiendo la capacidad de proliferación de adipocitos y comprometiendo el almacenaje de lípidos, lo que desencadenaría procesos de lipotoxicidad y un riesgo aumentado de desarrollar mecanismos de resistencia a la insulina, que podría derivar en la manifestación de síndrome metabólico.

Estos mecanismos descritos convierten a miR-483 en un candidato interesante a estudiar en la especie porcina, en la que ciertas razas y poblaciones han mostrado una marcada tendencia a desarrollar procesos de resistencia a la insulina o diabetes tipo II en respuesta a la ingesta de alimentos[135]. La descripción, identificación y anotación de este microRNA en el genoma porcino podría ayudar a establecer un mayor entendimiento de los procesos reguladores subyacentes en el desarrollo de patologías asociadas al metabolismo de lípidos, abriendo el camino a nuevos estudios sobre la regulación e interacción de los microRNAs en respuesta a la ingesta de alimentos durante la cría y posterior ciclo productivo del ganado porcino.

4. Conclusiones

El desarrollo del proyecto de Trabajo Fin de Máster se ha llevado a cabo siguiendo la mayoría de los puntos inicialmente planteados, y se ha expandido el trabajo de clasificación e identificación mediante un proceso de contraste por homología respecto a la anotación de microRNAs en humano. Durante el tiempo que ha durado este trabajo, se ha podido definir un proceso completo de detección y evaluación de nuevos candidatos pre-miRNAs, extensible al análisis en otras especies no modelo, mediante la comparación por homología, o mediante la detección *ab-initio* (trabajos que exceden el contenido de este proyecto). Gracias a la elaboración de este trabajo, se han podido adquirir nuevos conocimientos en el campo de la bioinformática y el Machine Learning aplicado al análisis de secuencias como los microRNAs, lo que supone y supondrá una oportunidad para la formación y campos de investigación a abordar en un futuro.

Seguidamente se presentan las principales conclusiones extraídas de los resultados obtenidos:

La representación de secuencias microRNAs mediante atributos estructurales cuantitativos ha sido una tarea compleja y no exenta de controversia en los últimos años. Diferentes aproximaciones han sido desarrolladas integrando multitud de atributos estructurales y estadísticos diferentes, fundamentalmente basados en las características de plegamiento de las secuencias en cuestión. Pese a los esfuerzos en identificar una serie reducida de atributos estructurales con gran poder discriminante entre estructuras pre-miRNAs y aquellas que no lo son, no se ha podido llegar a un consenso evidente sobre cuáles deberían ser objeto de atención preferente. No obstante, se ha podido constatar que el contenido en G+C, que influye notoriamente sobre el proceso de plegamiento y apareamiento de bases, constituye una variable importante respecto a las cualidades estructurales características de los microRNAs.

Las funciones y programas desarrollados para integrar todo el proceso, fundamentadas en el entorno de lenguaje R, así como en scripting BASH, podrán contribuir a la generalización del cálculo e identificación de secuencias a diversas especies no modelo, siendo el único requisito la definición de sets de datos positivos y negativos adecuados para el entrenamiento de los modelos de clasificación.

La predicción de estructuras pre-miRNA en diferentes genomas ha sido abordada en estudios previos mediante aproximaciones y técnicas diversas, entre ellas, las relacionadas con el uso de clasificadores basados en *Machine Learning.* Numerosos algoritmos han sido evaluados para esta tarea, siendo los algoritmos *Support Vector Machine* (SVM) y *Random Forest* (RF) unos de los más utilizados en la literatura. La precisión de los clasificadores entrenados ha podido ser optimizada hasta conseguir niveles superiores al 90% de poder predictor en muchos casos. En el proceso descrito en este trabajo hemos alcanzado niveles predictivos del 100% de eficacia para el clasificador SVM entrenado, habiéndose contrastado el proceso de entrenamiento con datos obtenidos para humano y porcino, así como con datos negativos publicados en

trabajos previos. No obstante, se debe ser cautos respecto a la elaboración de los sets de datos de entrenamiento debido a la enorme influencia que estos tendrán durante el entrenamiento y posterior clasificación de las secuencias. Una tarea no abordada en profundidad, debido a las limitaciones temporales, podría ser la revisión del proceso de selección de secuencias negativas, con el objetivo de que éstas sean lo suficientemente realistas como para generalizar el proceso de clasificación a otros casos particulares.

El uso de especies modelo con anotaciones fiables y extensas para detectar estructuras homólogas en otros genomas con anotaciones menos fiables y más incipientes para microRNAs, como es el caso del porcino, nos ha permitido identificar una serie de candidatos fiables a ser considerados para estudios posteriores de confirmación de su presencia como genes miRNA expresados a nivel celular y con funciones biológicas determinadas. Pese a ello, es evidente la limitación que plantea este modelo respecto a la detección y anotación de estructuras pre-miRNAs especie-específicas, así como secuencias filogenéticamente relacionadas pero que hayan sido sometidas a procesos de selección más intensos, con una conservación de secuencia menos evidente. Sin embargo, cabe destacar la posible adaptación del proceso de identificación desarrollado para la predicción ab-initio, no basado en homologías. Para ello se recomienda hacer uso de archivos de secuenciación *smallRNA-seq* generados en experimentos de expresión en diferentes tejidos, con el objetivo de filtrar las secuencias expresadas para extraer sus atributos estructurales y evaluar su posible clasificación como estructuras pre-miRNAs o no mediante el algoritmo SVM previamente entrenado.

El proceso de identificación desarrollado mediante detección por homología podría ser ampliado al uso de otras especies modelo como el ratón (*Mus musculus*). Además, las secuencias detectadas deberían ser confirmadas a nivel biológico mediante el desarrollo de paneles de expresión en diferentes tejidos, con el objetivo de confirmar su presencia mediante técnicas de RT-qPCR con sondas *Taqman* (ThermoFisher Scientific), así como sus posibles interacciones biológicas con mRNAs diana mediante experimentos de interferencia y detección en cultivos celulares. Por otra parte, se puede extender la detección de nuevos candidatos microRNA mediante un abordaje *ab-initio,* sin la limitación impuesta por la anotación de referencia homóloga para la especie modelo, en este caso el ser humano.

El estudio de las funciones biológicas e interacciones reguladoras desplegadas por el microRNA candidato ssc-miR-483 en porcino, resultará de interés para avanzar en el conocimiento sobre los determinantes genéticos y metabólicos relacionados con la obesidad, el depósito de lípidos y la susceptibilidad a enfermedades asociadas a alteraciones en el desarrollo debido a desequilibrios en la dieta en el ganado porcino.

5. Glosario

ARN: Ácido ribonucleico
ADN: Ácido desoxiribonucleico
miRNAs: micro-ARNs
mRNAs: ARNs mensajeros
pre-miRNA: precursor de micro-ARN
pri-miRNA: precursor de pre-micro-ARN
RISC: *RNA-induced silencing complex*
SVM: *Support Vector Machine*
RF: *Random Forest*
HMM: *hidden Markov Models*
MFE: *mínimum free energy*
Small-RNAseq: técnica de secuenciación de ARNs pequeños
SRA: Archivo de secuencias y metadatos (*Sequence Read Archive*)
FASTA: Archivo de secuencias
Mock-miRNA: secuencias artificiales creadas a partir de miRNAs
MMH: *Maximum Margin Hyperplane*
Mb: Megabases
BED: Archivo de posiciones genómicas

6. Bibliografía

1. Lewis, B. P., Burge, C. B. & Bartel, D. P. Conserved Seed Pairing, Often Flanked by Adenosines, Indicates that Thousands of Human Genes are MicroRNA Targets. *Cell* **120,** 15–20 (2005).
2. Lim, L. P., Glasner, M. E., Yekta, S., Burge, C. B. & Bartel, D. P. Vertebrate MicroRNA Genes. *Science (80-.).* **299,** 1540–1540 (2003).
3. Ambros, V. The functions of animal microRNAs. *Nature* **431,** 350–355 (2004).
4. Garzon, R., Fabbri, M., Cimmino, A., Calin, G. A. & Croce, C. M. MicroRNA expression and function in cancer. *Trends Mol. Med.* **12,** 580–7 (2006).
5. Price, N. L., Ramírez, C. M. & Fernández-Hernando, C. Relevance of microRNA in metabolic diseases. *Crit. Rev. Clin. Lab. Sci.* **51,** 305–320 (2014).
6. Jevsinek Skok, D. *et al.* Genome-wide *in silico* screening for microRNA genetic variability in livestock species. *Anim. Genet.* **44,** 669–677 (2013).
7. Bartel, D. P. MicroRNAs: genomics, biogenesis, mechanism, and function. *Cell* **116,** 281–97 (2004).
8. Lin, S. & Gregory, R. I. MicroRNA biogenesis pathways in cancer. *Nat. Rev. Cancer* **15,** 321–333 (2015).
9. Tessitore, A. *et al.* Therapeutic Use of MicroRNAs in Cancer. *Anticancer. Agents Med. Chem.* **16,** 7–19 (2016).
10. Sohel, M. H. Extracellular/Circulating MicroRNAs: Release Mechanisms, Functions and Challenges. *Achiev. Life Sci.* **10,** 175–186 (2016).
11. Axtell, M. J., Westholm, J. O. & Lai, E. C. Vive la différence: biogenesis and evolution of microRNAs in plants and animals. *Genome Biol.* **12,** 221 (2011).
12. Rodriguez, A., Griffiths-Jones, S., Ashurst, J. L. & Bradley, A. Identification of Mammalian microRNA Host Genes and Transcription Units. *Genome Res.* **14,** 1902–1910 (2004).
13. Kim, Y.-K. & Kim, V. N. Processing of intronic microRNAs. *EMBO J.* **26,** 775–783 (2007).
14. Lee, Y. *et al.* MicroRNA genes are transcribed by RNA polymerase II. *EMBO J.* **23,** 4051–4060 (2004).
15. CAI, X., Hagedorn, C. H. & Cullen, B. R. Human microRNAs are processed from capped, polyadenylated transcripts that can also function as mRNAs. *RNA* **10,** 1957–1966 (2004).
16. Denli, A. M., Tops, B. B. J., Plasterk, R. H. A., Ketting, R. F. & Hannon, G. J. Processing of primary microRNAs by the Microprocessor complex. *Nature* **432,** 231–235 (2004).
17. Filippov, V., Solovyev, V., Filippova, M. & Gill, S. S. A novel type of RNase III family proteins in eukaryotes. *Gene* **245,** 213–21 (2000).
18. Gregory, R. I., Chendrimada, T. P. & Shiekhattar, R. MicroRNA Biogenesis: Isolation and Characterization of the Microprocessor Complex. in *MicroRNA Protocols* **342,** 33–48 (Humana Press, 2006).
19. Yi, R., Qin, Y., Macara, I. G. & Cullen, B. R. Exportin-5 mediates the nuclear export of pre-microRNAs and short hairpin RNAs. *Genes Dev.* **17,** 3011–3016 (2003).
20. Zeng, Y. & Cullen, B. R. Structural requirements for pre-microRNA

binding and nuclear export by Exportin 5. *Nucleic Acids Res.* **32,** 4776–4785 (2004).
21. Bohnsack, M. T., Czaplinski, K. & Gorlich, D. Exportin 5 is a RanGTP-dependent dsRNA-binding protein that mediates nuclear export of pre-miRNAs. *RNA* **10,** 185–91 (2004).
22. LUND, E. & DAHLBERG, J. E. Substrate Selectivity of Exportin 5 and Dicer in the Biogenesis of MicroRNAs. *Cold Spring Harb. Symp. Quant. Biol.* **71,** 59–66 (2006).
23. Park, J.-E. *et al.* Dicer recognizes the 5′ end of RNA for efficient and accurate processing. *Nature* **475,** 201–205 (2011).
24. Gregory, R. I., Chendrimada, T. P., Cooch, N. & Shiekhattar, R. Human RISC Couples MicroRNA Biogenesis and Posttranscriptional Gene Silencing. *Cell* **123,** 631–640 (2005).
25. Rand, T. A., Petersen, S., Du, F. & Wang, X. Argonaute2 Cleaves the Anti-Guide Strand of siRNA during RISC Activation. *Cell* **123,** 621–629 (2005).
26. Chendrimada, T. P. *et al.* TRBP recruits the Dicer complex to Ago2 for microRNA processing and gene silencing. *Nature* **436,** 740–4 (2005).
27. Macfarlane, L.-A. & Murphy, P. R. MicroRNA: Biogenesis, Function and Role in Cancer. *Curr. Genomics* **11,** 537–61 (2010).
28. Ellwanger, D. C., Büttner, F. A., Mewes, H.-W. & Stümpflen, V. The sufficient minimal set of miRNA seed types. *Bioinformatics* **27,** 1346–50 (2011).
29. Lewis, B. P., Shih, I., Jones-Rhoades, M. W., Bartel, D. P. & Burge, C. B. Prediction of mammalian microRNA targets. *Cell* **115,** 787–98 (2003).
30. Jonas, S. & Izaurralde, E. Towards a molecular understanding of microRNA-mediated gene silencing. *Nat. Rev. Genet.* **16,** 421–433 (2015).
31. Bartel, D. P. MicroRNAs: Target Recognition and Regulatory Functions. *Cell* **136,** 215–233 (2009).
32. Thomas, M., Lieberman, J. & Lal, A. Desperately seeking microRNA targets. *Nat. Struct. Mol. Biol.* **17,** 1169–1174 (2010).
33. Akhtar, M. M., Micolucci, L., Islam, M. S., Olivieri, F. & Procopio, A. D. Bioinformatic tools for microRNA dissection. *Nucleic Acids Res.* **44,** 24–44 (2016).
34. Eiring, A. M. *et al.* miR-328 Functions as an RNA Decoy to Modulate hnRNP E2 Regulation of mRNA Translation in Leukemic Blasts. *Cell* **140,** 652–665 (2010).
35. Kim, D. H., Saetrom, P., Snove, O. & Rossi, J. J. MicroRNA-directed transcriptional gene silencing in mammalian cells. *Proc. Natl. Acad. Sci.* **105,** 16230–16235 (2008).
36. Vasudevan, S., Tong, Y. & Steitz, J. A. Switching from Repression to Activation: MicroRNAs Can Up-Regulate Translation. *Science (80-.).* **318,** 1931–1934 (2007).
37. Lau, N. C., Lim, L. P., Weinstein, E. G. & Bartel, D. P. An Abundant Class of Tiny RNAs with Probable Regulatory Roles in Caenorhabditis elegans. *Science (80-.).* **294,** 858–862 (2001).
38. Lee, R. C. & Ambros, V. An Extensive Class of Small RNAs in Caenorhabditis elegans. *Science (80-.).* **294,** 862–864 (2001).
39. Lim, L. P. *et al.* The microRNAs of Caenorhabditis elegans. *Genes Dev.*

17, 991–1008 (2003).
40. Tanzer, A. & Stadler, P. F. Molecular Evolution of a MicroRNA Cluster. *J. Mol. Biol.* **339,** 327–335 (2004).
41. Peterson, K. J., Dietrich, M. R. & McPeek, M. A. MicroRNAs and metazoan macroevolution: insights into canalization, complexity, and the Cambrian explosion. *BioEssays* **31,** 736–747 (2009).
42. Wheeler, B. M. *et al.* The deep evolution of metazoan microRNAs. *Evol. Dev.* **11,** 50–68 (2009).
43. Paps, J. & Hui, J. The phylogenetic utility and functional constraint of microRNA flanking sequences. *Proc. Biol. Sci.*
44. Aravin, A. A. *et al.* The small RNA profile during Drosophila melanogaster development. *Dev. Cell* **5,** 337–50 (2003).
45. Chen, P. Y. *et al.* The developmental miRNA profiles of zebrafish as determined by small RNA cloning. *Genes Dev.* **19,** 1288–1293 (2005).
46. Reinhart, B. J. *et al.* The 21-nucleotide let-7 RNA regulates developmental timing in Caenorhabditis elegans. *Nature* **403,** 901–906 (2000).
47. Sempere, L. F., Sokol, N. S., Dubrovsky, E. B., Berger, E. M. & Ambros, V. Temporal regulation of microRNA expression in Drosophila melanogaster mediated by hormonal signals and broad-Complex gene activity. *Dev. Biol.* **259,** 9–18 (2003).
48. Watanabe, T. *et al.* Stage-specific expression of microRNAs during *Xenopus* development. *FEBS Lett.* **579,** 318–324 (2005).
49. Ason, B. *et al.* Differences in vertebrate microRNA expression. *Proc. Natl. Acad. Sci.* **103,** 14385–14389 (2006).
50. Schulman, B. R. M., Esquela-Kerscher, A. & Slack, F. J. Reciprocal expression of *lin - 41* and the microRNAs *let - 7* and *mir - 125* during mouse embryogenesis. *Dev. Dyn.* **234,** 1046–1054 (2005).
51. Chen, J.-F. *et al.* The role of microRNA-1 and microRNA-133 in skeletal muscle proliferation and differentiation. *Nat. Genet.* **38,** 228–33 (2006).
52. Kalsotra, A. *et al.* The Mef2 Transcription Network Is Disrupted in Myotonic Dystrophy Heart Tissue, Dramatically Altering miRNA and mRNA Expression. *Cell Rep.* **6,** 336–345 (2014).
53. Sokol, N. S. & Ambros, V. Mesodermally expressed Drosophila microRNA-1 is regulated by Twist and is required in muscles during larval growth. *Genes Dev.* **19,** 2343–2354 (2005).
54. Kwon, C., Han, Z., Olson, E. N. & Srivastava, D. MicroRNA1 influences cardiac differentiation in Drosophila and regulates Notch signaling. *Proc. Natl. Acad. Sci.* **102,** 18986–18991 (2005).
55. Bentwich, I. *et al.* Identification of hundreds of conserved and nonconserved human microRNAs. *Nat. Genet.* **37,** 766–770 (2005).
56. Siomi, H. & Siomi, M. C. Posttranscriptional Regulation of MicroRNA Biogenesis in Animals. *Mol. Cell* **38,** 323–332 (2010).
57. Li, S.-C., Shiau, C.-K. & Lin, W.-C. Vir-Mir db: prediction of viral microRNA candidate hairpins. *Nucleic Acids Res.* **36,** D184-9 (2008).
58. Kozomara, A. & Griffiths-Jones, S. miRBase: annotating high confidence microRNAs using deep sequencing data. *Nucleic Acids Res.* **42,** D68–D73 (2014).
59. Nelson, P. T. *et al.* RAKE and LNA-ISH reveal microRNA expression and localization in archival human brain. *RNA* **12,** 187–91 (2006).

60. Bar, M. *et al.* MicroRNA Discovery and Profiling in Human Embryonic Stem Cells by Deep Sequencing of Small RNA Libraries. *Stem Cells* **26,** 2496–2505 (2008).
61. Janssen, S., Schudoma, C., Steger, G. & Giegerich, R. Lost in folding space? Comparing four variants of the thermodynamic model for RNA secondary structure prediction. *BMC Bioinformatics* **12,** 429 (2011).
62. Lorenz, R. *et al.* ViennaRNA Package 2.0. *Algorithms Mol. Biol.* **6,** 26 (2011).
63. Markham, N. R. & Zuker, M. UNAFold: software for nucleic acid folding and hybridization. *Methods Mol. Biol.* **453,** 3–31 (2008).
64. Lindow, M. & Gorodkin, J. Principles and Limitations of Computational MicroRNA Gene and Target Finding. *DNA Cell Biol.* **26,** 339–351 (2007).
65. Paczynska, P., Grzemski, A. & Szydlowski, M. Distribution of miRNA genes in the pig genome. *BMC Genet.* **16,** 6 (2015).
66. Lai, E. C., Tomancak, P., Williams, R. W. & Rubin, G. M. Computational identification of Drosophila microRNA genes. *Genome Biol.* **4,** R42 (2003).
67. Hertel, J. & Stadler, P. F. Hairpins in a Haystack: recognizing microRNA precursors in comparative genomics data. *Bioinformatics* **22,** e197–e202 (2006).
68. Gerlach, D., Kriventseva, E. V, Rahman, N., Vejnar, C. E. & Zdobnov, E. M. miROrtho: computational survey of microRNA genes. *Nucleic Acids Res.* **37,** D111-7 (2009).
69. Nam, J.-W., Kim, J., Kim, S.-K. & Zhang, B.-T. ProMiR II: a web server for the probabilistic prediction of clustered, nonclustered, conserved and nonconserved microRNAs. *Nucleic Acids Res.* **34,** W455-8 (2006).
70. Berezikov, E. *et al.* Phylogenetic Shadowing and Computational Identification of Human microRNA Genes. *Cell* **120,** 21–24 (2005).
71. Brameier, M. & Wiuf, C. Ab initio identification of human microRNAs based on structure motifs. *BMC Bioinformatics* **8,** 478 (2007).
72. Allmer, J. & Yousef, M. Computational methods for ab initio detection of microRNAs. *Front. Genet.* **3,** 209 (2012).
73. Huang, T.-H. *et al.* MiRFinder: an improved approach and software implementation for genome-wide fast microRNA precursor scans. *BMC Bioinformatics* **8,** 341 (2007).
74. Batuwita, R. & Palade, V. microPred: effective classification of pre-miRNAs for human miRNA gene prediction. *Bioinformatics* **25,** 989–995 (2009).
75. Nam, J.-W. *et al.* Human microRNA prediction through a probabilistic co-learning model of sequence and structure. *Nucleic Acids Res.* **33,** 3570–81 (2005).
76. Terai, G., Komori, T., Asai, K. & Kin, T. miRRim: a novel system to find conserved miRNAs with high sensitivity and specificity. *RNA* **13,** 2081–90 (2007).
77. Oulas, A. *et al.* Prediction of novel microRNA genes in cancer-associated genomic regions--a combined computational and experimental approach. *Nucleic Acids Res.* **37,** 3276–87 (2009).
78. Kadri, S., Hinman, V. & Benos, P. V. HHMMiR: efficient de novo prediction of microRNAs using hierarchical hidden Markov models. *BMC Bioinformatics* **10 Suppl 1,** S35 (2009).

79. Jiang, P. *et al.* MiPred: classification of real and pseudo microRNA precursors using random forest prediction model with combined features. *Nucleic Acids Res.* **35,** W339-44 (2007).
80. Zou, Q., Mao, Y., Hu, L., Wu, Y. & Ji, Z. miRClassify: An advanced web server for miRNA family classification and annotation. *Comput. Biol. Med.* **45,** 157–160 (2014).
81. Yousef, M. *et al.* Combining multi-species genomic data for microRNA identification using a Naive Bayes classifier. *Bioinformatics* **22,** 1325–1334 (2006).
82. Ritchie, W., Gao, D. & Rasko, J. E. J. Defining and providing robust controls for microRNA prediction. *Bioinformatics* **28,** 1058–1061 (2012).
83. Saçar, M. D., Hamzeiy, H. & Allmer, J. Can MiRBase Provide Positive Data for Machine Learning for the Detection of MiRNA Hairpins? *J. Integr. Bioinform.* **10,** 1–11 (2013).
84. Zerbino, D. R. *et al.* Ensembl 2018. *Nucleic Acids Res.* **46,** D754–D761 (2018).
85. Fromm, B. *et al.* MirGeneDB2.0: the curated microRNA Gene Database. *bioRxiv* 258749 (2018). doi:10.1101/258749
86. Chou, C.-H. *et al.* miRTarBase update 2018: a resource for experimentally validated microRNA-target interactions. *Nucleic Acids Res.* **46,** D296–D302 (2018).
87. Xue, C. *et al.* Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine. *BMC Bioinformatics* **6,** 310 (2005).
88. DP, B. MicroRNAs: genomics, biogenesis, mechanism, and function. *Cell* **116,** 281–297 (2004).
89. Yousef, M., Jung, S., Showe, L. C. & Showe, M. K. Learning from positive examples when the negative class is undetermined- microRNA gene identification. *Algorithms Mol. Biol.* **3,** 2 (2008).
90. Pruitt, K. D., Tatusova, T. & Maglott, D. R. NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Res.* **33,** D501–D504 (2004).
91. Longadge, R. & Dongre, S. Class Imbalance Problem in Data Mining Review. (2013).
92. Guo, X., Yin, Y., Dong, C., Yang, G. & Zhou, G. On the Class Imbalance Problem. in *2008 Fourth International Conference on Natural Computation* 192–201 (IEEE, 2008). doi:10.1109/ICNC.2008.871
93. Qazi, N. & Raza, K. Effect of Feature Selection, SMOTE and under Sampling on Class Imbalance Classification. in *2012 UKSim 14th International Conference on Computer Modelling and Simulation* 145–150 (IEEE, 2012). doi:10.1109/UKSim.2012.116
94. Wasikowski, M. & Chen, X. Combating the Small Sample Class Imbalance Problem Using Feature Selection. *IEEE Trans. Knowl. Data Eng.* **22,** 1388–1400 (2010).
95. Stegmayer, G., Yones, C., Kamenetzky, L., Macchiaroli, N. & Milone, D. H. Computational Prediction of Novel miRNAs from Genome-Wide Data. in *Methods in molecular biology (Clifton, N.J.)* **1654,** 29–37 (2017).
96. Yones, C., Stegmayer, G., Milone, D. H. & Sahinalp, C. Genome-wide pre-miRNA discovery from few labeled examples. *Bioinformatics* **34,** 541–549 (2018).

97. Pfeffer, S. *et al.* Identification of microRNAs of the herpesvirus family. *Nat. Methods* **2,** 269–276 (2005).
98. Ng, K. L. S. & Mishra, S. K. De novo SVM classification of precursor microRNAs from genomic pseudo hairpins using global and intrinsic folding measures. *Bioinformatics* **23,** 1321–1330 (2007).
99. Schultes, E. A., Hraber, P. T. & LaBean, T. H. Estimating the contributions of selection and self-organization in RNA secondary structure. *J. Mol. Evol.* **49,** 76–83 (1999).
100. Seffens, W. & Digby, D. mRNAs have greater negative folding free energies than shuffled or codon choice randomized sequences. *Nucleic Acids Res.* **27,** 1578–1584 (1999).
101. Freyhult, E., Gardner, P. P. & Moulton, V. A comparison of RNA folding measures. *BMC Bioinformatics* **6,** 241 (2005).
102. Moulton, V., Zuker, M., Steel, M., Pointon, R. & Penny, D. Metrics on RNA Secondary Structures. *J. Comput. Biol.* **7,** 277–292 (2000).
103. Fera, D. *et al.* RAG: RNA-As-Graphs web resource. *BMC Bioinformatics* **5,** 88 (2004).
104. Yones, C. A., Stegmayer, G., Kamenetzky, L. & Milone, D. H. miRNAfe: A comprehensive tool for feature extraction in microRNA prediction. *Biosystems* **138,** 1–5 (2015).
105. Quiles Sotillo, A. & Hevia Méndez, M. L. *Producción porcina intensiva*. (Agrícola Española, 2004).
106. OECD & Nations, F. and A. O. of the U. *OECD-FAO Agricultural Outlook 2017-2026*. (OECD Publishing, 2017). doi:10.1787/agr_outlook-2017-en
107. Saçar Demirci, M. D. & Allmer, J. Delineating the impact of machine learning elements in pre-microRNA detection. *PeerJ* **5,** e3131 (2017).
108. Chen, C. *et al.* Real-time quantification of microRNAs by stem-loop RT-PCR. *Nucleic Acids Res.* **33,** e179 (2005).
109. Witkos, T. M., Koscianska, E. & Krzyzosiak, W. J. Practical Aspects of microRNA Target Prediction. *Curr. Mol. Med.* **11,** 93–109 (2011).
110. Friedman, R. C., Farh, K. K.-H., Burge, C. B. & Bartel, D. P. Most mammalian mRNAs are conserved targets of microRNAs. *Genome Res.* **19,** 92–105 (2008).
111. Andres-Leon, E., Gonzalez Pena, D., Gomez-Lopez, G. & Pisano, D. G. miRGate: a curated database of human, mouse and rat miRNA-mRNA targets. *Database* **2015,** bav035-bav035 (2015).
112. Stothard P. The sequence manipulation suite: JavaScript programs for analyzing and formatting protein and DNA sequences. *Biotechniques* **28,** 1102–1104 (2000).
113. Kozlowski, P., Starega-Roslan, J., Legacz, M., Magnus, M. & Krzyzosiak, W. J. Structures of MicroRNA Precursors. in *Current Perspectives in microRNAs (miRNA)* 1–16 (Springer Netherlands, 2008). doi:10.1007/978-1-4020-8533-8_1
114. Jiang, M., Anderson, J., Gillespie, J. & Mayne, M. uShuffle: A useful tool for shuffling biological sequences while preserving the k-let counts. *BMC Bioinformatics* **9,** 192 (2008).
115. Huber, W. *et al.* Orchestrating high-throughput genomic analysis with Bioconductor. *Nat. Methods* **12,** 115–121 (2015).
116. Kuhn, M. caret: Classification and Regression Training. R package version 6.0-79. (2018).

117. Steinwart, I. & Christman, A. *Support Vector Machines*. (Springer New York, 2008). doi:10.1007/978-0-387-77242-4
118. Kohavi, R. & Kohavi, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. 1137--1143 (1995).
119. Langmead, B. Aligning short sequencing reads with Bowtie. *Curr. Protoc. Bioinforma.* **Chapter 11,** Unit 11.7 (2010).
120. Quinlan, A. R. & Hall, I. M. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26,** 841–2 (2010).
121. Tam, S., Tsao, M.-S. & McPherson, J. D. Optimization of miRNA-seq data preprocessing. *Brief. Bioinform.* **16,** 950–963 (2015).
122. Ziemann, M., Kaspi, A. & El-Osta, A. Evaluation of microRNA alignment techniques. *RNA* **22,** 1120–1138 (2016).
123. Durinck, S., Spellman, P. T., Birney, E. & Huber, W. Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. *Nat. Protoc.* **4,** 1184–91 (2009).
124. Neph, S. *et al.* BEDOPS: high-performance genomic feature operations. *Bioinformatics* **28,** 1919–1920 (2012).
125. de ON Lopes, I., Schliep, A. & de LF de Carvalho, A. C. The discriminant power of RNA features for pre-miRNA recognition. *BMC Bioinformatics* **15,** 124 (2014).
126. Vu, T. H. *et al.* Loss of imprinting of IGF2 sense and antisense transcripts in Wilms' tumor. *Cancer Res.* **63,** 1900–5 (2003).
127. Hao, J., Zhang, S., Zhou, Y., Hu, X. & Shao, C. MicroRNA 483-3p suppresses the expression of DPC4/Smad4 in pancreatic cancer. *FEBS Lett.* **585,** 207–213 (2011).
128. Ma, J. *et al.* miR-483-3p plays an oncogenic role in esophageal squamous cell carcinoma by targeting tumor suppressor EI24. *Cell Biol. Int.* **40,** 448–455 (2016).
129. ABUE, M. *et al.* Circulating miR-483-3p and miR-21 is highly expressed in plasma of pancreatic cancer. *Int. J. Oncol.* **46,** 539–547 (2015).
130. Perge, P. *et al.* Exosomal hsa-miR-483-5p and hsa-miR-101 are potential minimally invasive biomarkers of adrenocortical carcinoma. *Endocr. Abstr.* (2017). doi:10.1530/endoabs.49.GP3
131. Song, Q. *et al.* miR-483-5p promotes invasion and metastasis of lung adenocarcinoma by targeting RhoGDI1 and ALCAM. *Cancer Res.* **74,** 3031–42 (2014).
132. Niu, Z.-Y., Li, W.-L., Jiang, D.-L., Li, Y.-S. & Xie, X.-J. Mir-483 inhibits colon cancer cell proliferation and migration by targeting TRAF1. *Kaohsiung J. Med. Sci.* (2018). doi:10.1016/j.kjms.2018.04.005
133. Liu, M. *et al.* The IGF2 intronic miR-483 selectively enhances transcription from IGF2 fetal promoters and enhances tumorigenesis. *Genes Dev.* **27,** 2543–8 (2013).
134. Ferland-McCollough, D. *et al.* Programming of adipose tissue miR-483-3p and GDF-3 expression by maternal diet in type 2 diabetes. *Cell Death Differ.* **19,** 1003–12 (2012).
135. Torres-Rovira, L. *et al.* Diet-Induced Swine Model with Obesity/Leptin Resistance for the Study of Metabolic Syndrome and Type 2 Diabetes. *Sci. World J.* **2012,** 1–8 (2012).

7. Anexos

Scripts eMIRNA:

```
require(seqinr)
require(LncFinder)
require(stringr)
require(Biobase)
require(purrr)
require(data.table)
require(scales)
require(class)
require(kernlab)
require(caret)
require(e1071)

eMIRNA.Filter.by.Size <- function(file, prefix, a, b){
  setwd("~/")
  Dir <- "Sequence_FilterSize_Results"
  dir.create(file.path(Dir), showWarnings = FALSE)
  workdir <- "~/Sequence_FilterSize_Results/"
  setwd(workdir)

  #Checking Sequences and Filtering by size (50 to 150 nt)
  File0 <- unlist(lapply(file, readLines))
  n0 <- length(File0)
  ID0 <- File0[seq(1,n0,2)]
  Sequence0 <- File0[seq(2,n0,2)]
  Sequence0.split <- strsplit(Sequence0, "")
  Sequence0.length <- sapply(Sequence0.split, function(x) length(x))
  Index0.filter <- which(Sequence0.length >= a & Sequence0.length <= b)
  ID0.filter <- ID0[Index0.filter]
  Sequence0.filter <- Sequence0[Index0.filter]
  File0.filter.size <- c(rbind(ID0.filter, Sequence0.filter))
  name <- paste0(prefix, "_filter_size.fa")
  write(File0.filter.size, name)

  unlink("*.ps", recursive=T)

}

eMIRNA.Filter.by.Structure <- function(file, prefix){
  setwd("~/")
  Dir <- "Sequence_FilterStructure_Results"
  dir.create(file.path(Dir), showWarnings = FALSE)
  workdir <- "~/Sequence_FilterStructure_Results/"
  setwd(workdir)

  #Checking sequences and filtering by n loops
  File0 <- unlist(lapply(file, readLines))
  n0 <- length(File0)
  ID0 <- File0[seq(1,n0,2)]
  Sequence0 <- File0[seq(2,n0,2)]
  command1 <- paste0("RNAfold --MEA -d2 -p --noPS -i ", file)
  RNAfold1 <- system(command1, intern=TRUE)
  n <- length(RNAfold1)
  SecondaryStrc1 <- RNAfold1[seq(3,n,7)]
  SecondaryStrc <- gsub( " .*$", "", SecondaryStrc1)
  Nloop1 <- strsplit(SecondaryStrc, "\\((?=(\\.+\\)))", perl = TRUE)
  Nloop <- listLen(Nloop1) - 1
  Index0.nloop <- which(Nloop == 1)
  ID0.nloop <- ID0[Index0.nloop]
  Sequence0.nloop <- Sequence0[Index0.nloop]
  File0.filter.nloop <- c(rbind(ID0.nloop, Sequence0.nloop))
  name <- paste0(prefix, "_filter_nloop.fa")
  write(File0.filter.nloop, name)

  unlink("*.ps", recursive=T)

}
```

```
eMIRNA.Features <- function(path, file, prefix, Pval=FALSE){
  file_path <- paste0(path, file)
  setwd("~/")
  Dir <- "Sequence_Feature_Results"
  dir.create(file.path(Dir), showWarnings = FALSE)
  workdir <- "~/Sequence_Feature_Results/"
  setwd(workdir)
  message("eMIRNA.Features is calculating Structure Features for provided sequences. Please wait.")
  command1 <- paste0("RNAfold --MEA -d2 -p --noPS -i ", file_path)
  RNAfold1 <- system(command1, intern=TRUE)

  ##############################################
  ############ Sequence Features ############
  ##############################################

  #Triplet estimation from SVM-Triplet (Triplets)
  command2 <- paste0("RNAfold --noPS -i ", file_path, " > RNAfold_pred.txt")
  system(command2)
  command2.1 <- "1_check_query_content.pl RNAfold_pred.txt RNAfold_pred_checked.txt"
  system(command2.1)
  command2.2 <- "2_get_stemloop.pl RNAfold_pred_checked.txt RNAfold_pred_stemloop.txt 5"
  system(command2.2)
  command2.3 <- "3_step_triplet_coding_for_queries.pl RNAfold_pred_stemloop.txt Triplets.txt"
  system(command2.3)
  Triplets1 <- "Triplets.txt"
  Triplets1 <- unlist(lapply(Triplets1, readLines))
  Triplets.split1 <- strsplit(Triplets1, "\\s")
  Triplets1 <- as.numeric(as.character(unlist(Triplets.split1)))
  Triplets.along <- seq_along(Triplets1)
  Triplets <- split(Triplets1, ceiling(Triplets.along/32))
  Triplets <- do.call(rbind, Triplets)
  colnames(Triplets) <- c("1","2","3","4","5","6","7","8","9","10","11","12","13",
                          "14","15","16","17","18","19","20","21","22","23","24","25",
                          "26","27","28","29","30","31","32")
  T1 <- as.vector(Triplets[,1])
  T2 <- as.vector(Triplets[,2])
  T3 <- as.vector(Triplets[,3])
  T4 <- as.vector(Triplets[,4])
  T5 <- as.vector(Triplets[,5])
  T6 <- as.vector(Triplets[,6])
  T7 <- as.vector(Triplets[,7])
  T8 <- as.vector(Triplets[,8])
  T9 <- as.vector(Triplets[,9])
  T10 <- as.vector(Triplets[,10])
  T11 <- as.vector(Triplets[,11])
  T12 <- as.vector(Triplets[,12])
  T13 <- as.vector(Triplets[,13])
  T14 <- as.vector(Triplets[,14])
  T15 <- as.vector(Triplets[,15])
  T16 <- as.vector(Triplets[,16])
  T17 <- as.vector(Triplets[,17])
  T18 <- as.vector(Triplets[,18])
  T19 <- as.vector(Triplets[,19])
  T20 <- as.vector(Triplets[,20])
  T21 <- as.vector(Triplets[,21])
  T22 <- as.vector(Triplets[,22])
  T23 <- as.vector(Triplets[,23])
  T24 <- as.vector(Triplets[,24])
  T25 <- as.vector(Triplets[,25])
  T26 <- as.vector(Triplets[,26])
  T27 <- as.vector(Triplets[,27])
  T28 <- as.vector(Triplets[,28])
  T29 <- as.vector(Triplets[,29])
  T30 <- as.vector(Triplets[,30])
  T31 <- as.vector(Triplets[,31])
  T32 <- as.vector(Triplets[,32])

  StemF <- "RNAfold_pred_stemloop.txt"
  StemF <- unlist(lapply(StemF, readLines))
  StemF.ID <- grep(">", StemF, value=TRUE)
  StemF.ID <- gsub(">", "", StemF.ID)
  StemF.ID <- strsplit(StemF.ID, "_")
  StemF.ID <- lapply(StemF.ID, "[", -2)
  StemF.ID.index <- as.numeric(unlist(lapply(StemF.ID, "[", -2)))

  #ID
```

```
n <- length(RNAfold1)
ID <- RNAfold1[seq(1,n,7)]
ID <- gsub('>', '', ID)
ID <- ID[StemF.ID.index]

#Sequence
Sequence <- RNAfold1[seq(2,n,7)]
Sequence <- Sequence[StemF.ID.index]

#Length of Sequences (Length)
Length <- nchar(Sequence)

#G+C ratio (GC)
nG <- str_count(Sequence, "G")
nC <- str_count(Sequence, "C")
GC <- (nG + nC) / Length

#G/C ratio (G.Cr)
G.Cr <- nG / nC

#A+U/G+C ratio (AU.GCr)
nA <- str_count(Sequence, "A")
nU <- str_count(Sequence, "U")
AU.GCr <- (nA + nU) / (nG + nC)

#Base proportions Ratios (Ar, Ur, Gr, Cr)
Ar <- nA / Length
Ur <- nU / Length
Gr <- nG / Length
Cr <- nC / Length


#Dinucleotide Ratios (DNr)
AAr <- str_count(Sequence, "AA")/ Length
GGr <- str_count(Sequence, "GG")/ Length
CCr <- str_count(Sequence, "CC")/ Length
UUr <- str_count(Sequence, "UU")/ Length
AGr <- str_count(Sequence, "AG")/ Length
ACr <- str_count(Sequence, "AC")/ Length
AUr <- str_count(Sequence, "AU")/ Length
GAr <- str_count(Sequence, "GA")/ Length
GCr <- str_count(Sequence, "GC")/ Length
GUr <- str_count(Sequence, "GU")/ Length
CAr <- str_count(Sequence, "CA")/ Length
CGr <- str_count(Sequence, "CG")/ Length
CUr <- str_count(Sequence, "CU")/ Length
UAr <- str_count(Sequence, "UA")/ Length
UGr <- str_count(Sequence, "UG")/ Length
UCr <- str_count(Sequence, "UC")/ Length


#####################################################
########## Secondary Structure Features ##########
#####################################################

#RNAfold Secondary Structure (SecondaryStrc)
SecondaryStrc1 <- RNAfold1[seq(3,n,7)]
SecondaryStrc1 <- SecondaryStrc1[StemF.ID.index]
SecondaryStrc <- gsub( " .*$", "", SecondaryStrc1)

#Pairing Probabilities Structure (PairProbStrc)
PairProbStrc1 <- RNAfold1[seq(4,n,7)]
PairProbStrc1 <- PairProbStrc1[StemF.ID.index]
PairProbStrc <- gsub( " .*$", "", PairProbStrc1)

#RNAfold centroid structure (CentroidStrc)
CentroidStrc1 <- RNAfold1[seq(5,n,7)]
CentroidStrc1 <- CentroidStrc1[StemF.ID.index]
CentroidStrc <- gsub( " .*$", "", CentroidStrc1)

#Maximum Expected Accuracy Structure (MEAStrc)
MEAStrc1 <- RNAfold1[seq(6,n,7)]
MEAStrc1 <- MEAStrc1[StemF.ID.index]
MEAStrc <- gsub( " .*$", "", MEAStrc1)

#Hairpin length (Hl)
```

```
nl <- length(StemF)
Hl1 <- StemF[seq(3,nl,3)]
Hl2 <- strsplit(Hl1, "\\((?=(\\.+\\)))", perl = TRUE)
Hl1 <- lapply(Hl2, "[", -1)
Hl1 <- lapply(Hl1, function(x) gsub(").*", "", as.character(x)))
Hl3 <- lapply(Hl2, nchar)
nseq <- length(StemF.ID.index)
Hl3.2 <- as.list(rep(1, nseq))
Hl3 <- mapply('+', Hl3.2, Hl3, SIMPLIFY=FALSE)
Hl1 <- lapply(Hl1, nchar)
Hl <- as.vector(unlist(Hl1))

#5' and 3' Stem length (Steml5, Steml3)
Hl.list <- as.list(Hl)
list0 <- as.list(rep(0, nseq))
Steml1 <- mapply('c', list0, Hl.list, SIMPLIFY=FALSE)
Steml <- mapply('-', Hl3, Steml1, SIMPLIFY=FALSE)
Steml5 <- sapply(Steml, "[", 1)
Steml3 <- sapply(Steml, "[", 2)
Steml3 <- Steml3 - 1

#Number of basepairs in Secondary Structure (BP)
BP <- str_count(SecondaryStrc, "\\(")

#Number of basepairs in 5' and 3' Stem (BP5, BP3)
BP1 <- unlist(lapply(Hl2, "[", 1))
BP5 <- str_count(BP1, "\\(") + 1
BP2 <- unlist(lapply(Hl2, "[", 2))
BP2 <- sub("^\\.*", "", BP2)
BP3 <- str_count(BP2, "\\)")

#Number of mismatches in 5' and 3' Stem (Mism5, Mism3)
Mism5 <- str_count(BP1, "\\.")
Mism3 <- str_count(BP2, "\\.")

#Number of bulges in 5' and 3' Stem (Bulge5, Bulge3)
Bulge1 <- strsplit(BP1, "(\\.)+.")
Bulge5 <- listLen(Bulge1) - 1
Bulge2 <- strsplit(BP2, "(\\.)+.")
Bulge3 <- listLen(Bulge2) - 1

#Number of bulges by type (BulgeN1, BulgeN2, BulgeN3, BulgeN4, BulgeN5)
BulgeN1.5 <- str_count(BP1, "\\(\\.\\(")
BulgeN1.3 <- str_count(BP2, "\\)\\.\\)")
BulgeN2.5 <- str_count(BP1, "\\(\\.\\.\\(")
BulgeN2.3 <- str_count(BP2, "\\)\\.\\.\\)")
BulgeN3.5 <- str_count(BP1, "\\(\\.\\.\\.\\(")
BulgeN3.3 <- str_count(BP2, "\\)\\.\\.\\.\\)")
BulgeN4.5 <- str_count(BP1, "\\(\\.\\.\\.\\.\\(")
BulgeN4.3 <- str_count(BP2, "\\)\\.\\.\\.\\.\\)")
BulgeN5.5 <- str_count(BP1, "\\(\\.\\.\\.\\.\\.\\(")
BulgeN5.3 <- str_count(BP2, "\\)\\.\\.\\.\\.\\.\\)")

#Number of A-U, C-G and G-U pairs in 1 loop sequences (AUp, GCp, GUp)
Sequence2 <- strsplit(Sequence, "")
SecondaryStrc2 <- strsplit(SecondaryStrc, "")
Matches5 <- sapply(SecondaryStrc2, function(x) which(x == "("))
Matches3 <- sapply(SecondaryStrc2, function(x) which(x == ")"))
Extract5 <- mapply('[', Sequence2, Matches5)
Extract3 <- mapply('[', Sequence2, Matches3)
Extract3 <- sapply(Extract3, function(x) rev(x))
Pairs <- mapply('paste0', Extract5, Extract3, SIMPLIFY=FALSE)
CGc1 <- sapply(Pairs, function(x) sum(str_count(x, "CG")))
GCc1 <- sapply(Pairs, function(x) sum(str_count(x, "GC")))
GCp <- CGc1 + GCc1
AUc1 <- sapply(Pairs, function(x) sum(str_count(x, "AU")))
UAc1 <- sapply(Pairs, function(x) sum(str_count(x, "UA")))
AUp <- AUc1 + UAc1
GUc1 <- sapply(Pairs, function(x) sum(str_count(x, "GU")))
UGc1 <- sapply(Pairs, function(x) sum(str_count(x, "UG")))
GUp <- GUc1 + UGc1

Nloop1 <- strsplit(SecondaryStrc, "\\((?=(\\.+\\)))", perl = TRUE)
Nloop <- listLen(Nloop1) - 1

##################################################
```

```
######### Secondary Structure Statistics #########
##################################################



#Minimum Free Energy (MFE)
MFE <- as.numeric(regmatches(SecondaryStrc1,
                        gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                              SecondaryStrc1, perl=TRUE)))

#Ensemble Free Energy (EFE)
EFE <- as.numeric(regmatches(PairProbStrc1,
                        gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                              PairProbStrc1, perl=TRUE)))

#Centroid Free Energy (CFE)
CFE1 <- sub(".*? (.+)", "\\1", CentroidStrc1)
CFE1 <- gsub("\\{", "", CFE1)
CFE1 <- gsub("\\}", "", CFE1)
CFE1 <- gsub(" ", "", CFE1)
CFE1 <- unlist(strsplit(CFE1, "d="))
n2 <- length(CFE1)
CFE <- as.numeric(CFE1[seq(1,n2,2)])

#Centroid Distance to Ensemble (CDE)
CDE <- as.numeric(CFE1[seq(2,n2,2)])

#Maximum Expected Accuracy Structure Free Energy (MEAFE)
MEAFE1 <- sub(".*? (.+)", "\\1", MEAStrc1)
MEAFE1 <- gsub("\\{", "", MEAFE1)
MEAFE1 <- gsub("\\}", "", MEAFE1)
MEAFE1 <- gsub(" ", "", MEAFE1)
MEAFE1 <- unlist(strsplit(MEAFE1, "MEA="))
MEAFE <- as.numeric(MEAFE1[seq(1,n2,2)])

#Maximum Expected Accurary (MEA)
MEA <- as.numeric(MEAFE1[seq(2,n2,2)])

#Base pair Propensity (BPP)
BPP <- (BP / Length)

#Frequency of the MFE in ensemble (EFreq)
EFreq1 <- RNAfold1[seq(7,n,7)]
EFreq1 <- EFreq1[StemF.ID.index]
EFreq1 <- unlist(strsplit(EFreq1, ";"))
EFreq2 <- EFreq1[seq(1,n2,2)]
EFreq <- round(as.numeric(regmatches(EFreq2,
                        gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                              EFreq2, perl=TRUE))), 4)


#Ensemble Diversity (ED)
ED1 <- EFreq1[seq(2,n2,2)]
ED <- as.numeric(regmatches(ED1,
                        gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                              ED1, perl=TRUE)))

#Adjusted MFE (MFEadj)
MFEadj <- (MFE / Length)

#Adjusted EFE (EFEadj)
EFEadj <- (EFE / Length)

#Adjusted base pair Distance (Dadj)
Dadj <- (CDE / Length)

#Adjusted Shannon Entropy (SEadj)
SE <- -((Ar*log2(Ar))+(Ur*log2(Ur))+(Gr*log2(Gr))+(Cr*log2(Cr)))
SEadj <- (SE /Length)

#Difference between MFE and EFE Adjusted (DiffMFE.EFE)
DiffMFE.EFE <- ((MFE - EFE) / Length)

#Ratio between Adjusted MFE and GC (MFEadj.GC)
MFEadj.GC <- (MFEadj / GC)
```

```
#Ratio between Adjusted MFE and base pairs (MFEadj.BP)
MFEadj.BP <- (MFEadj / BP)

#dG
ID2 <- paste0(">", ID)
Fasta <- c(rbind(ID2, Sequence))
write.table(Fasta, "Fasta.fa", quote=F, row.names=F, col.names=F)
command5 <- paste0("melt.pl ", "Fasta.fa", " > Melt.txt")
system(command5)
Melt <- "Melt.txt"
Melt <- unlist(lapply(Melt, readLines))
n5 <- length(Melt)
Stats <- Melt[seq(nseq+2,n5,1)]
Stats <- unlist(strsplit(Stats, "\t"))
nStats <- length(Stats)
dG <- as.numeric(Stats[seq(1,nStats,4)])

#Adjusted dG (dGadj)
dGadj <- (dG / Length)

#dH
dH <- as.numeric(Stats[seq(2,nStats,4)])

#Adjusted dH (dHadj)
dHadj <- (dH / Length)

#dS
dS <- as.numeric(Stats[seq(3,nStats,4)])

#Adjusted dS (dSadj)
dSadj <- (dS / Length)

#dT
dT <- as.numeric(Stats[seq(4,nStats,4)])

#Adjusted dT (dTadj)
dTadj <- (dT / Length)


if (Pval == TRUE){
  #MFE P-value (MFE.Pval)
  seqs <- list()
  command2 <- list()
  for(i in 1:length(Sequence)){
    seqs[[i]] <- paste0("Ushuffle.exe -s ", Sequence[i], " -n 100 -k 2 -seed 1234")
  }
  seqs <- unlist(seqs)
    for(i in 1:length(seqs)){
      command2[[i]] <- system(seqs[[i]], intern=TRUE)
    }

  names(command2) <- paste0(ID, "_iterated")
  iterated_files <- list()
  for(i in seq_along(command2)) {
    write.table(command2[[i]], paste(names(command2)[i], ".txt", sep = ""),
                col.names = FALSE, row.names = FALSE, quote = FALSE)
    iterated_files[[i]] <- paste(names(command2)[[i]])
    iterated_files <- unlist(iterated_files)

  }

  iterated_files <- paste(iterated_files, ".txt", sep="")
  iterated_files <- paste0(iterated_files, collapse=" ")

  command3 <- paste0("cat ", iterated_files)
  iterated_seqs <- system(command3, intern=TRUE)
  write.table(iterated_seqs, "~/Sequence_Feature_Results/iterated_seqs.fa", row.names=F, col.names=F, quote=F)

  command4 <- "RNAfold --MEA -d2 -p --noPS -i iterated_seqs.fa"
  RNAfold2 <- system(command4, intern=TRUE)
  n4 <- length(RNAfold2)
  MFEit1 <- RNAfold2[seq(2,n4,6)]
  MFEit <- as.numeric(regmatches(MFEit1,
                          gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                                   MFEit1, perl=TRUE)))
  N <- 100
```

```
MFEit.along <- seq_along(MFEit)
MFEit.split <- split(MFEit, ceiling(MFEit.along/N))
R <- as.vector(sapply(Map(`<=`, MFEit.split, MFE), sum))
MFE.Pval <- (R / (N + 1))

#EFE P-value (EFE.Pval)
EFEit1 <- RNAfold2[seq(3,n4,6)]
EFEit <- as.numeric(regmatches(EFEit1,
                    gregexpr("(?>-)*[[:digit:]]+\\.*[[:digit:]]*",
                          EFEit1, perl=TRUE)))

EFEit.along <- seq_along(EFEit)
EFEit.split <- split(EFEit, ceiling(EFEit.along/N))
R2 <- as.vector(sapply(Map(`<=`, EFEit.split, EFE), sum))
EFE.Pval <- (R2 / (N + 1))

#Adjusted MFE Z-score (MFEz)
Lengthit <- rep(Length, each=100)
MFEitadj <- MFEit / Lengthit
MFEitadj.along <- seq_along(MFEitadj)
MFEitadj.split <- split(MFEitadj, ceiling(MFEitadj.along/N))
MFEitadj.mean <- as.vector(unlist(lapply(MFEitadj.split, mean)))
MFEitadj.sd <- as.vector(unlist(lapply(MFEitadj.split, sd)))
MFEz <- round(((MFEadj - MFEitadj.mean) / MFEitadj.sd), 4)

#MFEz P-value (MFEz.Pval)
MFEz.Pval <- 2*pnorm(-abs(MFEz))

#Adjusted EFE Z-score (EFEz)
EFEitadj <- EFEit / Lengthit
EFEitadj.along <- seq_along(EFEitadj)
EFEitadj.split <- split(EFEitadj, ceiling(EFEitadj.along/N))
EFEitadj.mean <- as.vector(unlist(lapply(EFEitadj.split, mean)))
EFEitadj.sd <- as.vector(unlist(lapply(EFEitadj.split, sd)))
EFEz <- ((EFEadj - EFEitadj.mean) / EFEitadj.sd)

#EFEz P-value (EFEz.Pval)
EFEz.Pval <- 2*pnorm(-abs(EFEz))

#Adjusted base pairing Propensity Z-score (BPPz)
BPPit1 <- RNAfold2[seq(2,n4,6)]
BPPit <- gsub( " .*$", "", BPPit1)
npairsit <- str_count(BPPit, "\\(")
BPPitadj <- npairsit / Lengthit
BPPitadj.along <- seq_along(BPPitadj)
BPPitadj.split <- split(BPPitadj, ceiling(BPPitadj.along/N))
BPPitadj.mean <- as.vector(unlist(lapply(BPPitadj.split, mean)))
BPPitadj.sd <- as.vector(unlist(lapply(BPPitadj.split, sd)))
BPPz <- ((BPP - BPPitadj.mean) / BPPitadj.sd)

#BPPz P-value (BPPz.Pval)
BPPz.Pval <- 2*pnorm(-abs(BPPz))

#Adjusted base pair Distance Z-score (Dz)
Dit1 <- RNAfold2[seq(4,n4,6)]
CFEit1 <- sub(".*? (.+)", "\\1", Dit1)
CFEit1 <- gsub("\\{", "", CFEit1)
CFEit1 <- gsub("\\}", "", CFEit1)
CFEit1 <- gsub(" ", "", CFEit1)
CFEit1 <- unlist(strsplit(CFEit1, "d="))
nit <- length(CFEit1)
CDEit <- as.numeric(CFEit1[seq(2,nit,2)])
Ditadj <- CDEit / Lengthit
Ditadj.along <- seq_along(Ditadj)
Ditadj.split <- split(Ditadj, ceiling(Ditadj.along/N))
Ditadj.mean <- as.vector(unlist(lapply(Ditadj.split, mean)))
Ditadj.sd <- as.vector(unlist(lapply(Ditadj.split, sd)))
Dz <- ((Dadj - Ditadj.mean) / Ditadj.sd)

#Dz P-value (Dz.Pval)
Dz.Pval <- 2*pnorm(-abs(Dz))


table <- as.data.frame(cbind(T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11, T12, T13, T14,
                             T15, T16, T17, T18, T19, T20, T21, T22, T23, T24, T25,
                             T26, T27, T28, T29, T30, T31, T32, EFreq, BPP, Dadj, SEadj,
```

```
                        DiffMFE.EFE, MFEz.Pval, EFE.Pval,  MFE.Pval,
                        EFEz.Pval, BPPz.Pval, Dz.Pval, Length, G.Cr, AU.GCr,
                        Ar, Ur, Gr, Cr, AAr, GGr, CCr, UUr, AGr, ACr, AUr,
                        GAr, GCr, GUr, CAr, CGr, CUr, UAr, UGr, UCr, Hl, Steml5,
                        Steml3, BP, BP5, BP3, Mism5, Mism3, Bulge5, Bulge3, BulgeN1.5,
                        BulgeN1.3, BulgeN2.5, BulgeN2.3, BulgeN3.5, BulgeN3.3,
                        BulgeN4.5, BulgeN4.3, BulgeN5.5, BulgeN5.3, AUp, GCp, GUp, MFE,
                        MFEadj, EFE, EFEadj, CFE, CDE, MEAFE, MEA, ED, MFEadj.GC, MFEadj.BP,
                        dG, dGadj, dH, dHadj, dS, dSadj, dT, dTadj, MFEz, EFEz, BPPz, Dz))


    table_1 <- table[, 1:43]
    table_2 <- table[, 44:112]
    table_2 <- as.data.frame(lapply(table_2, rescale))
    table <- cbind(table_1, table_2)
    rownames(table) <- ID
    colnames(table) <- seq(1:112)
    table[is.na(table)] <- 0


    final_table <- paste0(workdir, prefix, ".csv")
    write.table(table, final_table, sep=",", quote=F, col.names=NA)

    return(table)


    unlink("*.ps")
    unlink("*.txt")
    unlink("*fa*")

  } else {

    table <- as.data.frame(cbind(T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11, T12, T13, T14,
                        T15, T16, T17, T18, T19, T20, T21, T22, T23, T24, T25,
                        T26, T27, T28, T29, T30, T31, T32, EFreq, BPP, Dadj, SEadj,
                        DiffMFE.EFE, Length, G.Cr, AU.GCr,
                        Ar, Ur, Gr, Cr, AAr, GGr, CCr, UUr, AGr, ACr, AUr,
                        GAr, GCr, GUr, CAr, CGr, CUr, UAr, UGr, UCr, Hl, Steml5,
                        Steml3, BP, BP5, BP3, Mism5, Mism3, Bulge5, Bulge3, BulgeN1.5,
                        BulgeN1.3, BulgeN2.5, BulgeN2.3, BulgeN3.5, BulgeN3.3,
                        BulgeN4.5, BulgeN4.3, BulgeN5.5, BulgeN5.3, AUp, GCp, GUp, MFE,
                        MFEadj, EFE, EFEadj, CFE, CDE, MEAFE, MEA,
                        ED, MFEadj.GC, MFEadj.BP, dG, dGadj, dH, dHadj, dS, dSadj, dT, dTadj))

    unlink("*.ps")
    unlink("*.txt")
    unlink("*fa*")

    table_1 <- table[, 1:37]
    table_2 <- table[, 38:102]
    table_2 <- as.data.frame(lapply(table_2, rescale))
    table <- cbind(table_1, table_2)
    rownames(table) <- ID
    #colnames(table) <- seq(1:102)
    table[is.na(table)] <- 0

    final_table <- paste0(workdir, prefix, ".csv")
    write.table(table, final_table, sep=",", quote=F, col.names=NA)

    return(table)

  }

}


eMIRNA.Train <- function(pos, neg){
  final.table <- rbind(pos, neg)

  class <- c(rep("miRNA", nrow(pos)), rep("Other", nrow(neg)))

  final.table <- as.data.frame(cbind(final.table, class))
  final.table$class <- factor(final.table$class)


  set.seed(1234)
```

```
  intrain <- createDataPartition(y = final.table$class, p= 0.8, list = FALSE)
  training <- final.table[intrain,]
  testing <- final.table[-intrain,]


  #SVM Linear
  grid_lineal <- expand.grid(C = seq(0.01, 1, 0.1))

  trctrl <- trainControl(method = "cv", number = 10,
                         savePredictions = TRUE, classProbs=TRUE)

  svm_Linear <- train(class ~., data = training, method = "svmLinear",
                      trControl = trctrl,
                      tuneGrid = grid_lineal,
                      tuneLength = 10)

  return(svm_Linear)

}

eMIRNA.Predict <- function(model, features, prefix){
  pred <- predict(model, newdata= features)

  index <- which(pred == "miRNA")

  pred <- as.data.frame(rownames(features[index,]))
  colnames(pred) <- "Predicted_miRNAs"

  setwd("~/")
  Dir <- "Prediction_Results"
  dir.create(file.path(Dir), showWarnings = FALSE)
  workdir <- "~/Prediction_Results/"
  setwd(workdir)

  pred.path <- paste0(workdir, prefix, ".txt")
  write.table(pred, pred.path, quote=F, col.names=F, row.names=F)

  return(pred)

}


#! usr/bin/sh
## eMIRNA-Hunter

read -p 'Insert PATH to FASTA file: ' FASTA
read -p 'Insert PATH to output: ' OUT
read -p 'Insert PATH to Bowtie Index Genome Reference: ' REF
read -p 'Insert PREFIX output name: ' X

#Alignig miRNA sequences with Reference Genome Bowtie Index
bowtie -f -p 16 -n 1 -l 10 -m 100 -k 1 --best --strata $REF $FASTA -S $OUT$X.sam 2>$OUT$X.log

#Filtering aligned sequencies in SAM
awk '{FS="\t"}{OFS="\t"} $2 == 0 {print $1,$3,$4,$4+length($10)}' $OUT$X.sam > $OUT${X}_miRNAs_+.bed
awk '{FS="\t"}{OFS="\t"} $2 == 16 {print $1,$3,$4,$4+length($10)}' $OUT$X.sam > $OUT${X}_miRNAs_-.bed

# Adjusting positions
awk '/5p/{print $0}' $OUT${X}_miRNAs_-.bed > $OUT${X}_miRNAs_-_5p.bed
awk '/3p/{print $0}' $OUT${X}_miRNAs_-.bed > $OUT${X}_miRNAs_-_3p.bed
awk '/3p/{print $0}' $OUT${X}_miRNAs_+.bed > $OUT${X}_miRNAs_+_3p.bed
awk '/5p/{print $0}' $OUT${X}_miRNAs_+.bed > $OUT${X}_miRNAs_+_5p.bed
awk '!/5p|3p/{print $0}' $OUT${X}_miRNAs_-.bed > $OUT${X}_miRNAs_-_nobranch.bed
awk '!/5p|3p/{print $0}' $OUT${X}_miRNAs_+.bed > $OUT${X}_miRNAs_+_nobranch.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-50,$4+10,$1"_homologue","1","+"}' $OUT${X}_miRNAs_+_3p.bed >
$OUT${X}_miRNAs_+_3p_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-10,$4+50,$1"_homologue","1","+"}' $OUT${X}_miRNAs_+_5p.bed >
$OUT${X}_miRNAs_+_5p_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-50,$4+10,$1"_homologue","1","-"}' $OUT${X}_miRNAs_-_5p.bed >
$OUT${X}_miRNAs_-_5p_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-10,$4+50,$1"_homologue","1","-"}' $OUT${X}_miRNAs_-_3p.bed >
$OUT${X}_miRNAs_-_3p_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-50,$4+10,$1"-mock3p_homologue","1","+"}' $OUT${X}_miRNAs_+_nobranch.bed >
$OUT${X}_miRNAs_+_3p_branched_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-10,$4+50,$1"-mock5p_homologue","1","+"}' $OUT${X}_miRNAs_+_nobranch.bed >
$OUT${X}_miRNAs_+_5p_branched_final.bed
```

```
awk '{FS="\t"}{OFS="\t"}{print $2,$3-50,$4+10,$1"-mock5p_homologue","1","-"}' $OUT${X}_miRNAs_-_nobranch.bed > $OUT${X}_miRNAs_-_5p_branched_final.bed
awk '{FS="\t"}{OFS="\t"}{print $2,$3-10,$4+50,$1"-mock3p_homologue","1","-"}' $OUT${X}_miRNAs_-_nobranch.bed > $OUT${X}_miRNAs_-_3p_branched_final.bed

#Generating .bed
cat $OUT${X}_miRNAs_+_3p_final.bed $OUT${X}_miRNAs_+_5p_final.bed $OUT${X}_miRNAs_-_3p_final.bed $OUT${X}_miRNAs_-_5p_final.bed $OUT${X}_miRNAs_+_3p_branched_final.bed $OUT${X}_miRNAs_+_5p_branched_final.bed $OUT${X}_miRNAs_-_3p_branched_final.bed $OUT${X}_miRNAs_-_5p_branched_final.bed | sort -k1,1n -k2,2n > $OUT${X}_homolog_miRNAs.bed

#Generating FASTA file from .bed
bedtools getfasta -fi $REF.fa -bed $OUT${X}_homolog_miRNAs.bed -s -name -fo $OUT${X}_homolog_miRNAs.fa

rm $OUT${X}_miRNAs_+.bed
rm $OUT${X}_miRNAs_-.bed
rm $OUT${X}_miRNAs_-_5p.bed
rm $OUT${X}_miRNAs_-_3p.bed
rm $OUT${X}_miRNAs_+_5p.bed
rm $OUT${X}_miRNAs_+_3p.bed
rm $OUT${X}_miRNAs_+_3p_final.bed
rm $OUT${X}_miRNAs_+_5p_final.bed
rm $OUT${X}_miRNAs_-_3p_final.bed
rm $OUT${X}_miRNAs_-_5p_final.bed
rm $OUT${X}_miRNAs_-_nobranch.bed
rm $OUT${X}_miRNAs_+_nobranch.bed
rm $OUT${X}_miRNAs_+_3p_branched_final.bed
rm $OUT${X}_miRNAs_+_5p_branched_final.bed
rm $OUT${X}_miRNAs_-_3p_branched_final.bed
rm $OUT${X}_miRNAs_-_5p_branched_final.bed


#! usr/bin/sh
## eMIRNA-Seeker

read -p 'Insert PATH to GTF annotation: ' GTF
read -p 'Insert PATH to miRNA model annotation: ' miRNA
read -p 'Insert PATH to list of predicted miRNAs: ' PRED
read -p 'Insert PATH to homolog miRNAs positions file: ' HOM
read -p 'Insert PATH to output: ' OUT
read -p 'Insert PREFIX output name: ' X

#Prepare GTF reference
grep miRNA $GTF > $OUT${X}_temp1.gtf
awk '$3 == "gene" {print $0}' $OUT${X}_temp1.gtf > $OUT${X}_temp2.gtf
awk '{FS="\t"}{OFS="\t"}{print $1,$4,$5,$9,1,$7}' $OUT${X}_temp2.gtf > $OUT${X}_temp3.bed

# Generate .bed from predicted miRNAs
awk 'NR==FNR{c[$1]++;next};c[$4]>0' $PRED $HOM | sort -k1,1V -k2,2n > $OUT${X}_Predicted_miRNAs.bed

# Generate .bed for annotated and unannotated predicted miRNAs
bedtools intersect -wa -s -a $OUT${X}_Predicted_miRNAs.bed -b $OUT${X}_temp3.bed | sort -k1,1V -k2,2n > $OUT${X}_Predicted_miRNAs_annotated.bed
bedtools intersect -wa -s -v -a $OUT${X}_Predicted_miRNAs.bed -b $OUT${X}_temp3.bed | sort -k1,1V -k2,2n > $OUT${X}_Predicted_miRNAs_NO_annotated.bed

# Elongate positions in no annotated miRNAs for neighbourhood seeking
awk '{FS="\t"}{OFS="\t"}{if($2 > 2000000) print $1,$2-2000000,$3+2000000,$4; else print $1,1,$3+2000000,$4}' $OUT${X}_Predicted_miRNAs_NO_annotated.bed | sed 's/-mock3p//g' | sed 's/-mock5p//' > $OUT${X}_temp6.bed
sort-bed $OUT${X}_temp6.bed > $OUT${X}_temp7.bed
awk 'NR==FNR {a[$4]++} NR!=FNR && a[$4]==1' $OUT${X}_temp7.bed $OUT${X}_temp7.bed > $OUT${X}_temp7.2.bed

# Prepare reference model miRNA annotation
sed 's/;/\t/g' $miRNA | sed 's/Name=//g' | grep -v "#" | awk '{FS="\t"}{OFS="\t"}{print $1,$2,$3,$4,$5,$6,$7,$8,$11}' > $OUT${X}_temp9.bed
awk '{FS="\t"}{OFS="\t"}{if($3 == "miRNA") print $0}' $OUT${X}_temp9.bed | awk '{FS="\t"}{OFS="\t"}{print $1,$4,$5,$9}' | sed 's/chr//g' > $OUT${X}_temp10.bed

# Elongate positions in model miRNAs annotation for neighbourhood seeking
awk '{FS="\t"}{OFS="\t"}{if($2 > 2000000) print $1,$2-2000000,$3+2000000,$4; else print $1,1,$3+2000000,$4}' $OUT${X}_temp10.bed > $OUT${X}_temp11.bed
sort-bed $OUT${X}_temp11.bed > $OUT${X}_temp12.bed

# Contrast predicted unannotated miRNAs with model miRNA annotation
```

```
awk '{print $4}' $OUT${X}_temp7.2.bed | sed 's/_homologue//g' > $OUT${X}_temp13.txt
awk 'NR==FNR{c[$1]++;next};c[$4]>0' $OUT${X}_temp13.txt $OUT${X}_temp12.bed > $OUT${X}_temp14.bed
awk 'NR==FNR { a[$4]++ } NR!=FNR && a[$4]==1' $OUT${X}_temp14.bed $OUT${X}_temp14.bed > $OUT${X}_temp14.2.bed
sed 's/_homologue//g' $OUT${X}_temp7.2.bed > $OUT${X}_temp7.3.bed
awk 'NR==FNR{c[$4]++;next};c[$4]>0' $OUT${X}_temp14.2.bed $OUT${X}_temp7.3.bed > $OUT${X}_temp7.4.bed

# Extract genes in predicted miRNAs neighbourhood and in model miRNA annotation
Rscript script_biomaRt_calc.R $OUT $X
sed 's/_homologue//g' $OUT${X}_temp15.txt | sed 's/RF[0-9]*,//g' | sed 's/,RF[0-9]*//g' | sed 's/,,/,/g' > $OUT${X}_temp17.txt

# Count matching genes in neighbourhoods
awk -v s="," '{OFS="\t"} NR==FNR {a[$1]=s $2 s; next} {c=0; n=split($2,b,s); for(i=1;i<=n;i++) c+=(a[$1] ~ s b[i] s); print $1,c}' $OUT${X}_temp16.txt $OUT${X}_temp17.txt > $OUT${X}_temp18.txt

# Select predicted miRNAs with common model miRNA neighbourhood
awk '{FS="\t"}{OFS="\t"}{if($2 > 1) print $1}' $OUT${X}_temp18.txt | sed 's/r/R/g' > $OUT${X}_temp19.txt

# Count gene number in interval
awk -F '[\t,]' '{print $1, NF-1}' $OUT${X}_temp17.txt | sed 's/r/R/g' | sed 's/ /\t/g' > $OUT${X}_temp19.2.txt
paste $OUT${X}_temp18.txt $OUT${X}_temp19.2.txt | awk '{FS=OFS="\t"}{print $1,$2,$4}' > $OUT${X}_temp19.3.txt
awk '{FS="\t"}{OFS="\t"}{if($2 > 1) print $1,$2/$3}' $OUT${X}_temp19.3.txt > $OUT${X}_temp19.4.txt

# Extract predicted miRNAs with coincidential neighbouring genes
awk '{FS="\t"}{OFS="\t"}{print $1,$2,$3,$4}' $OUT${X}_Predicted_miRNAs_NO_annotated.bed | sed 's/-mock3p//g' | sed 's/-mock5p//g' | sed 's/_homologue//g' > $OUT${X}_temp20.txt
echo -e "Chr\tStart\tEnd\tmiRNA\tNeighbourhood_Score" > $OUT${X}_temp21.txt
awk 'NR==FNR{c[$1]++;next};c[$4]>0' $OUT${X}_temp19.4.txt $OUT${X}_temp20.txt > $OUT${X}_temp22.txt
sort $OUT${X}_temp19.4.txt > $OUT${X}_temp23.txt
sort -k4,4 $OUT${X}_temp22.txt > $OUT${X}_temp24.txt
paste $OUT${X}_temp23.txt $OUT${X}_temp24.txt | awk '{FS=OFS="\t"}{print $3,$4,$5,$6,$2}' | sort -k1,1V -k2,2n -k5,5 > $OUT${X}_temp25.txt
cat $OUT${X}_temp21.txt $OUT${X}_temp25.txt > $OUT${X}_temp26.txt
sed 's/hsa/ssc/g' $OUT${X}_temp26.txt | sed 's/-5p//g' | sed 's/-3p//' > $OUT${X}_Feasable_predicted_miRNAs.txt

rm *temp*


## script_biomaRt_calc.R

# Biomart miRNA Neighbourhood query
library(biomaRt)
args <- commandArgs(trailingOnly = TRUE)

# data
file <- paste0(args[1], args[2], "_temp7.4.bed")
d <- read.table(file)

# specify the database
ensembl = useMart("ensembl", dataset = "sscrofa_gene_ensembl")

# loop through rows, get genes, then paste with collapse,
# and finally bind back with data d.
res <- cbind(
  d,
  genes = apply(d, 1, function(i){
    x <- getBM(attributes=c("external_gene_name"),
               filters = c("chromosome_name" , "start", "end"),
               values = list(i[1], i[2], i[3]),
               mart = ensembl)

    # return genes, comma separated
    paste(x$external_gene_name, collapse = ",")
  })
)

genes <- as.vector(res$genes)
miRNAs <- as.vector(res$V4)
Results <- as.data.frame(cbind(miRNAs, genes))

# data
file2 <- paste0(args[1], args[2], "_temp14.2.bed")
d2 <- read.table(file2)
```

```
# specify the database
ensembl = useMart("ensembl", dataset = "hsapiens_gene_ensembl")

# loop through rows, get genes, then paste with collapse,
# and finally bind back with data d.
res2 <- cbind(
  d2,
  genes = apply(d2, 1, function(i){
    x <- getBM(attributes=c("sscrofa_homolog_associated_gene_name"),
               filters = c("chromosome_name" , "start", "end"),
               values = list(i[1], i[2], i[3]),
               mart = ensembl)

    # return genes, comma separated
    paste(x$sscrofa_homolog_associated_gene_name, collapse = ",")
  })
)

genes2 <- as.vector(res2$genes)
miRNAs2 <- as.vector(res2$V4)
Results2 <- as.data.frame(cbind(miRNAs2, genes2))

out <- paste0(args[1], args[2], "_temp15.txt")
out2 <- paste0(args[1], args[2], "_temp16.txt")

write.table(Results, out, quote=F, sep="\t", row.names=F)
write.table(Results2, out2, quote=F, sep="\t", row.names=F)
```