UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

Facultad de Ingeniería Eléctrica

División de Estudios de Posgrado

ESTIMACIÓN Y PREDICCIÓN DEL ESTADO EMOCIONAL BASADO EN SEÑALES DE AUDIO

TESIS

Que para obtener el grado de

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

Sistemas Computacionales

Presenta

Bryan Eduardo Martínez Guzmán

Dr. Jaime Cerda Jacobo

Director de Tesis

Morelia, Michoacán Mayo 2018

Deseo expresar mis agradecimientos a:

A la Universidad Michoacana de San Nicolás de Hidalgo Facultad de Ingeniería Eléctrica (UMSNH) porque ha sido mi casa durante este largo lapso de tiempo en ella.

Un agradecimiento especialmente a mi madre Jackeline Guzmán Reyes, mi abuela Victoria Reyes Balverde por su apoyo incondicional, cariño y confianza brindada para llegar a lograr esta meta.

A mi asesor Dr. Jaime Cerda Jacobo por el apoyo brindado con el cual he podido culminar este trabajo de tesis.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por los apoyos financieros para cursar mis estudios de posgrado.

Lista de Publicaciones

- Bryan E. Martínez and Jaime Cerda Jacobo. An improved characterization methodology to efficiently deal with the speech emotion recognition problem. In *Power, Electronics and Computing (ROPEC 2017), IEEE International Autumn Meeting on*, pages 1-6

Resumen

En este trabajo de tesis se presenta una metodología para predecir el estado emocional de los seres humanos basados en señales de voz emitidas durante una conversación. La metodología expuesta se divide en dos escenarios i.e. clasificación y predicción del estado emocional. En el escenario de la clasificación, el método de caracterización desarrollado comienza con una señal de voz, a la cual son extraídas diferentes características para obtener cuatro vectores bi-dimensionales. Después de esto, se utiliza un esquema de clasificación de dos etapas. En la primera etapa por cada vector se buscan las mejores zonas que representan a cada estado emocional, aplicando un kernel a los vectores de características. En la segunda etapa se utilizan varios clasificadores estándar para evaluar las zonas emocionales antes encontradas. Ésta estrategia se manifiesta con buen desempeño en todos los clasificadores, salvo en el caso de las Máquinas de Soporte Vectorial, con una tasa de clasificación superior (91-100 %) respecto a varios trabajos en la literatura.

En el segundo escenario lo esencial es dar una estimación del estado emocional en cualquier instante, además de dar una predicción de las emociones que serán dominantes t- pasos adelante en el tiempo. Para resolver estas actividades en este trabajo de tesis se implementaron los Procesos de Decisión Parcialmente Observables de Markov. En estos procesos de decisión al obtener el conjunto de políticas y acciones predominantes en un tiempo se crea internamente una función que tiene las propiedades de ser lineal y convexa a pedazos. Para poder predecir que emoción o emociones serán dominantes, es necesario determinar la duración de las políticas sobre la función lineal convexa anterior. En la actualidad existen varios algoritmos que se encargan de determinar éstas regiones, los cuales se dividen en exactos y aproximados. En este trabajo de tesis se implementaron de ambos tipos, el motivo es que nos interesa el tiempo y la precisión que se obtiene en este problema de la predicción del estado emocional. La aportación de los algoritmos implementados es que se basan en señales de voz a diferencia de trabajos anteriores. En la literatura del reconocimiento de emociones se han evaluado a las personas por gesticulaciones faciales, manera de hablar, su cultura y otros factores.

Palabras clave Reconocimiento de emociones, Procesos de Decisión Parcialmente Observables de Markov, Zonas emocionales, Función lineal convexa a pedazos y Predicción del estado emocional.

Abstract

In this thesis work, a methodology is presented to predict the emotional state of human beings based on voice signals emitted during a conversation. The exposed methodology is divided into two scenarios i.e. classification and prediction of emotional state. In the classification scenario, the characterization method developed starts with a voice signal, to which different characteristics are extracted to obtain four bi-dimensional vectors. After this, a two-stage classification scheme is used. In the first stage for each vector, the best zones that represent each emotional state are searched by applying a kernel to the feature vectors. In the second stage, several standard classifiers are used to evaluate the emotional zones previously found. This strategy manifests itself with good performance in all the classifiers, except in the case of the Support Vector Machines, with a higher classification rate (91-100 %) with respect to several works in the literature.

In the second scenario, the essential thing is to give an estimate of the emotional state at any moment, in addition to giving a prediction of the emotions that will be dominant t- steps ahead in time. To solve these activities in this thesis work, Partially Observable Markov Decision Processes were implemented. In these decision processes, by obtaining the set of predominant policies and actions in a while, a function is created internally that has the properties of being piecewise convex linear. In order to predict which emotion or emotions will be dominant, it is necessary to determine the duration of the policies on the previous convex linear function. At present there are several algorithms that are responsible for determining these regions, which are divided into exact and approximate. In this thesis work were implemented of both types, the reason is that we are interested in the time and precision obtained in this problem of the prediction of the emotional state. The contribution of the algorithms implemented is that they are based on voice signals, unlike previous works. In the literature on the recognition of emotions, people have been evaluated by facial gestures, manner of speaking, their culture and other factors.

Keywords Emotion recognition, Partially Observable Markov Decision Processes, Emotional Zones, Piecewise convex linear function and Predicting emotional state.

Contenido

Lista de Figuras

Lista de Tablas

Lista de Algoritmos

Lista de Símbolos

Símbolos para la clasificación

a	Factor de descuento del filtro de pre-énfasis
$E(n)$	Energía de tiempo corto en el instante discreto n
$h(n)$	Filtro de pre-énfasis
$H(n)$	Entropía de tiempo corto en el instante discreto n
$H_S(n)$	Entropía espectral de tiempo corto en el instante discreto n
$R(n)$	Función de auto-correlación en el instante discreto n
$R_D(n)$	Función de diferencia de magnitudes en el instante discreto n
th	Valor de umbral para la segmentación
$w(n)$	Frame o marco de ventana
$\mathbf{X}$	Vector de características
$Z(n)$	Tasa de cruces por cero de tiempo corto en el instante discreto n

Símbolos para la predicción

$'$	El apóstrofe indica que se realizó una transición al siguiente estado
b	Estado de creencia
b_0	Estado de creencia inicial
b_t	Estado de creencia en el tiempo t
A	Conjunto acciones
R	Conjunto de recompensas
S	Conjunto de estados
T	Conjunto de transiciones entre estados
O	Conjunto de observaciones
Z	Conjunto de probabilidades entre observaciones
γ	Valor de descuento
$E[\cdot]$	Función del valor esperado
π	Definición de una política
π_t	Definición de una política en el tiempo t
$V_{\pi,t}(s)$	Suma esperada de las recompensas que se obtienen desde un estado inicial s ejecutando la política π para t pasos adelante. Ésta también se conoce cómo el árbol de políticas.

$Q^{\pi}(b,a)$	Función-Q la cual proporciona la suma del valor de descuento esperado de las recompensas, para una acción a en un estado de creencia b, con la política π
$\oplus$	Operador suma cruzada
$\setminus$	Operador substracción de conjuntos
Γ_n^a	Conjunto de vectores de políticas del espacio resumido
γ_n^a	Conjunto de vectores de políticas parsimonioso
η	Constante de normalización
$Q(b,a)$	Representa el conjunto mínimo de árboles de políticas
$\hat{Q}(b,a)$	Estimación de $Q(b,a)$
δ	Condición que se debe maximizar sujetos a que la recompensa es positiva
e_s	Vector unitario
S_o^a	Conjunto de vectores por estados, acciones y observaciones
$\emptyset$	Conjunto vacío
h	Historia de un árbol de búsquedas
G	Simulador de un POMDP con Monte Carlo
$N(s,a)$	Número de visitas de un nodo en el árbol de búsquedas Monte Carlo
c	Proporción de exploración
T	Nodo en el árbol de búsquedas Monte Carlo
$\sim$	Similar, es decir, se tiene una distribución normal de variables aleatorias con media μ y varianza σ.

Lista de Mnemónicos

Mnemónicos para la clasificación

ACF	Función de autocorrelación.
AMDF	Función de diferencias de promedio de magnitudes.
BES-DB	Base de datos emocional de Berlín.
DES-DB	Base de datos emocional Danés.
DT	Árboles de decisión.
EMO-DB	Base de datos emocional de Berlín.
ET	Arboles extra.
FN	Falsos negativos.
FP	Falsos positivos.
FPR	Régimen de positivos falsos.
FS	Selección de características.
GD-female	Dependiente de género femenino.
GD-male	Dependiente de género masculino.
HCI	Interacción Humano-Computadora.
LDC-DB	Base de datos del consorcio de datos lingüístico.
ln	Logaritmo natural.
MFCC	Coeficientes Cepstrales de MEL.
RF	Bosques aleatorios.
s	Señal.
SAVEE-DB	Base de datos de emoción expresada audiovisual.
SD	Dependiente del parlante.
SER	Reconocimiento de emociones a través de la voz.
sgn	Signo.
SI	Independiente del parlante.
SP	Procesamiento de voz.
SUSAS-DB	Base de datos del habla bajo estrés simulada y real.
TEO	Operadores de energía Teager.
TN	Positivos negativos.
TP	Positivos verdaderos.
TPR	Régimen de positivos verdaderos.
ZCR	Cruces por cero.

Mnemónicos para la predicción

ACP	Procesos de control afectivo.
EPA	Vector de tres valores para el estado afectivo.
MCTS	Árbol de búsqueda de Monte Carlo.
MDP	Procesos de decisión de Markov.
POMDP	Procesos de decisión parcialmente observables de Markov.
POMCP	POMDP con el método de Monte-Carlo.
SDS	Sistemas de diálogo hablado.
SDS-POMDP	POMDP aplicados a sistemas de diálogo hablado.
SVM	Maquinas con soporte vectorial.
UCT	Algoritmo para seleccionar las acciones en el árbol de búsqueda Monte Carlo.

Capítulo 1

Introducción

Las emociones juegan un papel importante en actividades diarias de los seres humanos. Estas actividades incluyen la toma de decisiones, relacionarse con otras personas, incluso hasta la manera de comunicar un mensaje, y cómo éste es percibido por una persona diferente. En este capítulo se describe el problema del reconocimiento de emociones a través de la voz en la interacción entre humanos.

1.1. Reconocimiento del estado emocional

En la actualidad uno de los temas más interesantes en la interacción Humano-Computadora es el reconocimiento de las emociones humanas. Esto se debe a, qué identificar las emociones de alguna persona que se encuentra en interacción con un sistema, puede crear un experiencia más natural para el ser humano. Por ejemplo, un videojuego que se pueda estar configurando a través de las partidas y con ello mostrar hacia el jugador el escenario con el que se encuentre más cómodo. Estos sistemas tienen una amplia cantidad de aplicaciones como juegos, servicios de llamadas inteligentes, o bien puede ayudar en los estudios clínicos de diagnóstico del estado emocional en pacientes psiquiátricos entre otros [Fadil et al., 2015]. Todas estas aplicaciones se han estado trabajando con gesticulaciones faciales y corporales, utilizando imágenes o generalmente con señales de voz [Fadil et al., 2015].

La base del reconocimiento emocional cae en el área del *Machine Learning* o *Aprendizaje de Máquina*. El aprendizaje de máquina es un campo de la informática que da a las

computadoras la capacidad de aprender con datos sin estar explícitamente programado, es decir, crear programas capaces de generalizar comportamientos a partir de una información suministrada en forma de ejemplos. Para la mayoría de las aplicaciones prácticas, las variables de entrada originales suelen pre-procesarse, para transformarlas en un nuevo espacio de variables donde, se espera el problema de reconocimiento de patrones será más fácil de resolver. Esto reduce en gran medida la variabilidad dentro de cada clase, lo que hace que sea mucho más fácil para un algoritmo de reconocimiento de patrones distinguir entre diferentes clases. La etapa de pre-procesamiento a veces también se denomina cómo *extracción de características* [Christopher, 2016].

Las aplicaciones en las que los datos de entrenamiento comprenden ejemplos de los vectores de entrada junto con sus correspondientes vectores objetivo se conocen como problemas de *aprendizaje supervisado*. El aprendizaje supervisado requiere de un profesor que mida el rendimiento, es decir, en éste aprendizaje se tienen variables de entrada y variables de salida, donde la tarea de la máquina de aprendizaje es utilizar un algoritmo que tiene una función que mapea desde las entradas a las salidas. La meta es aproximar la función de mapeo tan bien como se pueda mientras se tengan nuevas variables de entrada para predecir a las nuevas variables de salida [Duda et al., 2012]. Casos en el que el objetivo es asignar cada un vector de entrada a un número finito de categorías discretas, se denomina como *problema de clasificación*. Si el resultado deseado consiste en una o más variables continuas, entonces esa tarea se llama *problema de regresión* [Christopher, 2016].

En otros problemas de reconocimiento de patrones, los datos de entrenamiento consisten en un conjunto de vectores de entrada $\mathbf{X}$ sin ningún valor objetivo correspondiente. El objetivo en tales problemas de *aprendizaje no supervisados* puede ser descubrir grupos de ejemplos similares dentro de los datos, donde se llama *agrupamiento*, o determinar la distribución de datos dentro del espacio de entrada [Christopher, 2016]. Es decir, en el aprendizaje no supervisado no hay un entrenamiento explícito dónde se forman concentraciones naturales con los patrones de entrada. Estas concentraciones ó cúmulos se definen explícita o implícitamente en el sistema de agrupación en sí mismo. La diferencia principal de éste tipo de aprendizaje respecto al aprendizaje supervisado es que no hay algún tipo de conocimiento a priori [Duda et al., 2012].

1.2. Descripción general

En éste trabajo de tesis se presenta un sistema para estimación y predicción del estado emocional con las emociones i.e. Enojo, Aburrimiento, Disgusto, Ansiedad, Miedo, Felicidad, Tristeza, Sorpresa y Neutral. Éste sistema comienza a partir de audios a los cuales se extraen dos tipos de características i.e. energía y frecuencia, para obtener vectores de características $\mathbf{X}$ bi-dimensionales. La longitud de los vectores depende del número de los mejores puntos que representen cada emoción, después de haber utilizado un *kernel* donde se separan las muestras emocionales que representa cada estado emocional. De está manera se crean zonas emocionales, es decir, cada emocion tiene diferentes lugares donde puede estar presente. Estas zonas emocionales son utilizadas con diálogos a través de la voz entre parlantes.

En una conversación se extraen características y se mide la distancia por proximidad de los nuevos puntos a los centroides de las zonas emocionales. La emoción que se percibe en el instante de tiempo actual corresponde a la mayor concentración de puntos en alguna zona emocional. Al obtener está última emoción se mandan a llamar los Procesos de Decisión Parcialmente Observables de Markov aplicando diferentes técnicas de optimización para obtener una predicción de las emociones que serán dominantes algunos pasos adelante. Éste proceso se repite por cada nueva observación emocional que perciba el sistema.

1.3. Motivación de la tesis

Hoy en día existe una gran cantidad de software comercial que utiliza el reconocimiento de palabras por medio de la voz para una infinidad de aplicaciones, algunos ejemplos más populares son Siri de Apple, Cortana de Windows y el analizador de voz de Google. Todas estas aplicaciones se centran en reconocer las frases que se expresan y a partir de ello se puede buscar un domicilio con google maps, buscar archivos en la computadora hasta estar escribiendo sin presionar una tecla, entre más actividades que se pueden desarrollar solo con el reconocimiento de frases. Sin embargo, dados estos trabajos me surgen algunas inquietudes que se relacionan a ¿Cómo influyen las emociones en los diálogos?, ¿Qué tanta información se puede obtener de conversaciones emocionales? y finalmente, ¿Cómo se

podrían aprovechar esos parámetros emocionales en diferentes áreas?. Hasta este momento no existe un software comercial que utilice las emociones a través de la voz para dar respuesta. Entonces, la motivación de este trabajo de tesis se centra en encontrar los mejores parámetros fáciles de calcular para obtener una representación del estado emocional. Una vez que se obtiene ésta representación emocional, pueda ser aplicada en un sistema de tiempo real con el fin de que en cada interacción con algún usuario se gane información, y a partir de ello predecir qué emoción permanecerá durante un periodo de tiempo. Un sistema desarrollado con este enfoque puede tener varias aplicaciones:

- Suponiendo en las extorsiones telefónicas. Un sistema que detecte extorsiones bien puede ser configurado para detectar emociones cómo enojo, miedo y posiblemente neutral. Con ésta configuración del sistema, en caso de percibir alguna llamada telefónica con las emociones anteriores, se puede mandar una notificación a la policía cibernética dónde se menciona la existencia de alguna llamada con características emocionales sospechosas, para que se pueda monitorear y prevenir algún tipo de daño.

- En el área de la robótica se pueden crear sistemas en los cuales el robot pueda interactuar con el usuario dependiendo de su estado emocional y podría ser aplicado en zonas para atención a clientes, atención medica, sala de espera entre otros casos. Un sistema de éste tipo también puede servir para dar seguimiento a los pacientes con alteraciones emocionales tanto para su control y tratamiento.

- En el mundo de los videojuegos seria muy interesante la aplicación de un sistema cómo éste, especialmente por la cantidad de personas que se involucran en la actualidad. Supongamos un juego donde la configuración de cada nivel depende en gran medida del estado emocional del parlante, y debido a los cambios emocionales que se perciban por el micrófono se este modificando tanto la música de fondo, funciones de los jugadores y escenarios. Un enfoque de ésta manera es algo no muy común en la actualidad que daría una experiencia más real. Al igual en el mundo de los videojuegos se podrían prevenir accidentes, porqué existen casos donde los jugadores experimentan niveles tan altos de euforia al estar perdiendo la partida y comienzan con amenazas hacia otros jugadores. Todas éstas acciones han terminado en muerte. Con éste sistema se pueden

detectar estos niveles de euforia y se podrían prevenir varias de estas situaciones.

- Finalmente, un sistema con este enfoque puede tener otras aplicaciones sencillas cómo la manera de dejar comentarios de quejas y sugerencias. En este caso solo se puede hacer una grabación donde se evalúa el desempeño del servicio y dependiendo de las emociones clasificarlo como algo bueno, regular o malo.

1.4. Objetivos de la Tesis

El objetivo de este trabajo de tesis es implementar un sistema de reconocimiento de emociones basado en señales de voz, el cual pueda monitorear el estado emocional del parlante y además, tenga la capacidad de predecir algunos pasos adelante la o las emociones que predominan en un diálogo.

1.4.1. Objetivos particulares

- Implementar una caracterización con señales de voz, que al clasificar tenga un buen desempeño respecto a la literatura y que sea aplicable a un sistema de tiempo real.
- En caso de obtener multi-emociones a través de alguna conversación resaltar la emoción predominante.
- Obtener un mapa de distribución de emociones con las bases de datos más comunes.
- Predecir el tiempo promedio en la transición del estado emocional.
- Realizar una planificación precisa t-pasos adelante.

1.5. Contenido de la tesis

En el Capítulo 2 son presentados e.g. los antecedentes relacionados a los sistemas de reconocimiento de emociones a través de la voz, el método de clasificación implementado y las pruebas realizadas para la clasificación de las emociones con dos bases de datos con muestras emocionales de voz. El Capítulo 3 menciona los Procesos de Decisión de Markov

(MDP), seguidos a la vez por los MDP Parcialmente Observables (POMDP) y todos los componentes necesarios para poder utilizarlos e.g. estados, transiciones y observaciones que se obtienen de diálogos afectivos. El Capítulo 4 menciona acerca de algunas políticas de optimización, incluyendo a los POMDP con la optimización de Monte-Carlo mejor conocidos como POMCP. En éste Capítulo 4 también se describen algunas otras políticas de optimización, además de las estrategias que han surgido para el problema de la planificación a grandes horizontes. Las pruebas relacionadas a la predicción del estado emocional se presentan en éste Capítulo 4 las cuales tienen la finalidad de elegir el o los mejores parámetros para ser aplicados a un sistema de tiempo real. Finalmente, en el Capítulo 5 se presentan las conclusiones de este trabajo de tesis, así cómo el trabajo futuro que se puede realizar dentro del área de Cómputo Afectivo.

Capítulo 2

Clasificación del estado emocional

En el capítulo anterior se presentó el planteamiento del problema a resolver, el cual se divide en clasificación y predicción del estado emocional a través de la voz. En este capítulo se va a presentar la base para poder realizar una buena clasificación que es aplicable en sistemas de tiempo real. La finalidad del Capítulo 2 es proporcionar una descripción de los métodos que han sido utilizados para la clasificación de las emociones, así cómo describir algunas herramientas utilizadas para éste propósito e.g. bases de datos, extracción de características y modelos emocionales entre otras.

2.1. Introducción

Hoy en día el Procesamiento de Voz (Speech Processing, SP por sus siglas del inglés) tiene aplicaciones interesantes incluyendo Juegos Online, Aplicaciones Móviles, Centros de Llamadas Inteligentes (Smart Call Centers) y muchos otros donde existe al menos un método de comunicación Hombre-Máquina. Esto implica que la máquina puede reconocer y procesar señales de voz, sin embargo, una máquina no puede entender el estado emocional de los parlantes [El Ayadi et al., 2011]. El área del Reconocimiento de Emociones a través de la Voz (Speech Emotion Recognition, SER por sus siglas del inglés) trata de distinguir los estados emocionales. Sin embargo, distinguir los estados emocionales es una tarea muy difícil ya que cada parlante tiene propiedades diferentes e.g. estilo del habla, velocidad del habla y pronunciación (que dependen del parlante y de su entorno). Es de-

cir, los seres humanos tienen diferentes sistemas vocales y características emocionales que varían entre la juventud y la mediana edad, así cómo la voz entre los hombres y las mujeres [Ramakrishnan and El Emary, 2013]. Todas estas variaciones de los parlantes afectan directamente a los límites de cada emoción, por lo tanto, es un problema difícil evaluar qué parte de cada frase pertenece a cada estado emocional. Además, puede haber más de una emoción percibida en la misma expresión [El Ayadi et al., 2011]. El objetivo de SER es identificar las características que determinan las emociones del parlante basándose en la señal de voz [Zhang et al., 2017]. Los dos principales desafíos que enfrenta SER para el reconocimiento de emociones son: Primero, es cómo extraer características que pueden ser utilizadas para representar correctamente las emociones. Segundo, es cómo construir un algoritmo de aprendizaje que sea capaz de mapear lo mejor posible los datos de entrada a la salida. [Gu et al., 2016].

En un sistema SER se extraen varias características e.g. características prosódicas tambien conocidas cómo características del Pitch, características espectrales especialmente con los Coeficientes Cepstrales de Frecuencia de MEL (Mel Frequency Cepstral Coefficients, MFCC por sus siglas del inglés), características de intensidad (Energía), características no lineales con los Operadores de Energía Teager (Teager Energy Operator features, TEO por sus siglas del inglés [Li et al., 2012]), características bi-espectrales [Yaacob and Polat, 2017], características de forma de onda glotal y características de bio-coherencia [Yogesh et al., 2017]. Para la clasificación éstas características se toman por pares para representar un vector bi-dimensional con valores de activación y valencia. En este trabajo de tesis las características que se extraen tienen ésta forma de vectores bi-dimensionales.

La activación es la energía necesaria para expresar una emoción (es decir, la tristeza tiene poca energía mientras que la felicidad y el enojo tienen mucha energía). La valencia es un valor de la emoción (la felicidad es un valor positivo y la tristeza es un valor negativo) (ver Tabla 2.1) [Ververidis et al., 2004, Ramakrishnan and El Emary, 2013]. Las emociones se combinan posteriormente en clases más grandes definidas por activación alta/baja y valencia positiva/negativa [Pohjalainen and Alku, 2014]. En general estas características se relacionan más al comportamiento del sistema nervioso, presión sanguínea, pulmones, músculos y ritmo cardiaco [Hale, 2017]. Al igual las emociones también se des-

Tabla 2.1: Distribución de emociones por valores de activación y valencia

	Alta activación	*Baja activación*
Valencia negativa	enojo miedo	disgusto tristeza
Valencia positiva	alegria	aburrimiento

criben por las respuestas fisiológicas y comportamientos del parlante [Grimm et al., 2007, Ramakrishnan and El Emary, 2013]. El enojo es la emoción con la energía más alta así cómo el nivel de tono más alto. En el estado de enojo, el ritmo es rápido y las sílabas son acentuadas, pero la última palabra no suele acentuarse. El disgusto se expresa con un tono bajo y niveles de intensidad bajos. El estado emocional del miedo se relaciona a un nivel de tono alto y un nivel de intensidad elevado. Cuando alguna persona se encuentra en el estado de alegría, tanto el tono cómo la energía promedio son altos y la velocidad de la conversación también aumenta. Los niveles bajos de intensidad media y tono medio se miden cuando los sujetos expresan tristeza. La velocidad del habla bajo circunstancias similares es generalmente más lenta que en el estado neutral [Ramakrishnan and El Emary, 2013].

La Tabla 2.1 muestra una manera simple para clasificar las emociones por activación y valencia. Sin embargo, existen otras emociones que no se pueden asignar correctamente porqué se encuentran en los límites de la activación o valencia. Por ejemplo, la emoción neutral tiene valencia positiva pero por su energía no es fácil agruparla a una clase. Con lo anteriormente dicho, varios investigadores han llegado a clasificar las emociones básicas, es decir, al menos seis emociones parecen ser persistentes en todos los subconjuntos de la humanidad. Estas seis emociones i.e. Alegría, Enojo, Miedo, Sorpresa, Disgusto y Tristeza. Con estas emociones los humanos podemos expresar diversas combinaciones sutiles, contextuales, de combinación y pseudoemociones, por ejemplo, el sarcasmo, escepticismo, distanciamiento, confusión, compromiso, decepción, compasión, frustración y diversión entre otras [Hale, 2017]. En este trabajo de tesis se realizaron pruebas con al menos esas seis emociones básicas i.e. Alegría, Enojo, Miedo, Sorpresa, Disgusto y Tristeza.

El reconocimiento de las señales de voz para la Interacción Humano-Computadora (Human-Computer Interaction, HCI por sus siglas del inglés) tiene algunos pasos comunes

por diferentes autores que han estado trabajando en esta área: Primero, computar estadísticos de alto nivel a partir de las características prosódicas. Segundo, estimar los estadísticos globales de las características extraídas y usar un método de selección de características para reducir la dimensión. Por ejemplo, selección de características hacia adelante (Forward) o hacia atrás (Backward), algoritmos genéticos y algoritmos evolutivos [Yaacob and Polat, 2017, Ramakrishnan and El Emary, 2013]. El último paso consiste en emplear un clasificador para evaluar el rendimiento del sistema, ya que las características también deciden el clasificador apropiado [Ramakrishnan and El Emary, 2013, Zhang et al., 2017].

Existen algunas estrategias para evaluar el desempeño de estos sistemas de reconocimiento e.g. independiente del parlante (Speaker Independent, SI por sus siglas del inglés), dependiente del parlante (Speaker Dependent, SD por sus siglas del inglés), masculino dependiente de género (Genere Male Dependent, GD-male por sus siglas del inglés) y femenino dependiente del género (Genere Female Dependent, GD-female por sus siglas del inglés) [Yogesh et al., 2017]. En este trabajo de tesis la estrategia para evaluar el desempeño fue independiente del parlante (SI).

2.2. Bases de datos y extracción de características

Hay una variedad de bases de datos para SER que contienen una amplia cantidad de muestras emocionales en imágenes, videos y audios. De estas bases de datos las más utilizados son: Berlin Emotional Database (EMO-DB disponible en http://emodb.bilderbar.info/index-1024.html) también conocida cómo BES-DB en [Yogesh et al., 2017], Linguistic Data Consortium (LDC-DB) Emotional, Danish Emotional Database (DES-DB), Speech Under Simulated and Actual Stress (SUSAS-DB), Surrey Audio-Visual Expressed Emotion (SAVEE-DB disponible en http://kahlan.eps.surrey.ac.uk/savee/) y Chinese Academy of Sciences Emotional Speech. En [Ververidis and Kotropoulos, 2003, El Ayadi et al., 2011] existen algunas otras referencias. Estas bases de datos comparten emociones comunes e.g. Aburrimiento, Disgusto, Miedo, Alegría, Tristeza y Sorpresa para diferentes idiomas: Alemán, Danés, Inglés y Chino. Es importante tener en cuenta que no todas las bases de datos son públicas y gratuitas, por lo que se complica proporcionar un enlace para su descarga. Es conveniente

mencionar que las bases de datos emocionales se dividen en tres categorías las cuales son: Natural (espontánea con emociones reales), Simulada (producida por actores profesionales) y Elicitada (las emociones son inducidas por lo que el diálogo que se obtiene no es ni neutro ni simulado) [Zhang et al., 2017]. Todas estas características se mencionan porqué son variaciones que afectan a la clasificación de las emociones.

En el proceso de extracción de características es muy común por los autores que han estado trabajando en esta área obtener vectores con alta dimensionalidad, por éste motivo ellos utilizan un algoritmo Selector de Características. La finalidad de la Selección de Características (Feature Selection, FS por sus siglas del inglés) es elegir un subconjunto de las mejores características, que sea suficiente para alcanzar su representación y de esa manera reducir el costo de almacenamiento y ejecución en la clasificación/aprendizaje. Trabajar con problemas de alta dimensión puede causar un exceso de entrenamiento, lo que lleva a una degradación del rendimiento [Yogesh et al., 2017]. En este trabajo de tesis se restringe agregar algún algoritmo selector de características, a diferencia de los trabajos reportados en la literatura. En este trabajo de tesis se dedica más tiempo al procesamiento de la señal, porqué se conoce que las características principales de la voz humana es su estructura dinámica. Está estructura de la voz al ser muestreada se relaciona con series de tiempo de observaciones a las cuales se les pueden aplicar procesos de suavizados para descubrir patrones [Wagner et al., 2007].

El Pitch en las señales de voz esta relacionado a que tan alto o tan bajo es percibido el tono. Éste valor depende del número de vibraciones por segundo producido por las vocales. El Pitch también representa la principal correlación acústica entre el tono y la entonación [Ververidis et al., 2004, Mattheyses et al., 2006, Gerhard, 2003]. Por éste motivo, el Pitch es uno de los principales protagonistas en la literatura y es también uno de los principales utilizados en éste trabajo de tesis. Existen dos principales estrategias que son fáciles para obtener el Pitch i.e. la Función Autocorrelación (Autocorrelation Function, ACF por sus siglas del inglés) [Mattheyses et al., 2006] y la Función de Diferencia de Promedio de Magnitudes (Average Magnitude Difference Function, AMDF por sus siglas del inglés) [Gerhard, 2003]. La diferencia entre estas funciones es el tiempo de procesamiento, dado que ACF utiliza multiplicaciones mientras que AMDF usa sólo sumas. AMDF es

clave en el procesamiento de voz en tiempo real. Algunos autores limitan el Pitch entre 80 - 320 Hz [Ververidis et al., 2004, Mattheyses et al., 2006, Gerhard, 2003] ya que es allí donde se encuentran las principales características SER. Además, que la energía del Pitch se concentra en un determinado rango de frecuencia. Éste rango de frecuencia a su vez corresponde al área de audición más sensible, es posible que las diferencias perceptivas importantes entre varias clases de la voz probablemente también se manifiesten en esta región [Pohjalainen and Alku, 2014]. Los estudios psicológicos muestran que la prosodia (Pitch, intensidad y velocidad del habla) y la calidad de la voz son las más importantes para distinguir entre las emociones según la percepción humana. En general, el Pitch y la energía parecen estar correlacionados con la activación. La alta activación se relaciona a valores del Pitch y energía altos, mientras, una baja activación se relaciona a valores del Pitch y energía bajos [Ramakrishnan and El Emary, 2013].

Darekar y Dhande et al. [Darekar and Dhande, 2017] establecen que para obtener características con una alta precisión, la energía, MFCC y el Pitch pueden fusionarse entre sí. Esto quiere decir que el rendimiento del sistema se aumentará con los parámetros i.e Pitch, energía, frecuencia vocal, MFCC y formantes de frecuencias. Un formante es el pico de intensidad en el espectro de un sonido; se trata de una concentración de energía que se da en una determinada frecuencia. Entonces, con éste conjunto de parámetros se va a obtener una mejor eficiencia en la detección del estado emocional.

Las Tablas 2.2 y 2.3 contienen un resumen de la extracción de características y funciones que fueron implementadas en este trabajo de tesis. En estas tablas se utiliza la letra M para representar el tamaño de la señal, $s(m)$ para representar una muestra de la señal s en el instante m y $w(n)$ para representar el n-ésimo marco de ventana o frame w, donde N corresponde a la cantidad total de marcos de ventana que se obtienen. Un marco de ventana es un vector limitado de muestras de la señal original.

En la Tabla 2.2 $R(n)$ de la fila N^{o} 1 representa la función de auto-correlación de tiempo corto de tiempo discreto n y $R_D(n)$ de la fila N^{o} 2 representa la función de promedio de diferencia de magnitudes de tiempo corto de tiempo discreto n. Con estas dos ecuaciones se puede obtener el Pitch. $Z(n)$ de la fila N^{o} 3 representa la tasa de cruces por cero (Zero Crossing Rate, ZCR por sus siglas del inglés) de tiempo discreto n. Un cruce

Tabla 2.2: Tabla para extracción de características

Nº	*Nombre*	*Característica*
1	ACF	$R(n) = \sum_{m=-\infty}^{M-n-1} s(m)s(m+n)$ donde $s(m)$ es la muestra de la señal
2	AMDF	$R_D(n) = \sum_{m=-\infty}^{M-n-1} \lvert s(m) - s(m+n)\rvert$ donde $s(m)$ es la muestra de la señal
3	Tasa de cruces por cero de tiempo corto	$Z(n) = \sum_{m=-\infty}^{\infty} w(n-m) \lvert sgn\left[s(m)\right] - sgn\left[s(m-1)\right]\rvert$ donde: $sgn\left[s(n)\right] = 1, s(n) \geq 0$ $sgn\left[s(n)\right] = -1, s(n) < 0$ $w(n) = \frac{1}{2M}, 0 \leq n \leq M-1$ $w(n) = 0$, de otra manera
4	Energía	$E(n) = \sum_{m=n-M+1}^{n} s(m)^2$
5	Entropía	Los valores v_i ocurren en la señal x_i, de acuerdo a: $f_i = \sum_{j=1}^{M} \varphi\left(x_i, v_i\right)$ Donde la función $\varphi(x_i, v_i)$ regresa: 1 cuando $x_i = v_i$ 0 cuando $x_i \neq v_i$ Las probabilidades $p_i = \frac{f_i}{M}$ Sujetos a: $\sum_{i=1}^{n} p_i = 1$ Información propia $I(p_i) = ln\left(\frac{1}{p_i}\right) = -\ln\left(p_i\right)$ Entropía $H(n) = -\sum_{i=1}^{n} p_i \ln\left(p_i\right)$ donde: $H_{\text{mín}} = -\ln\left(1\right) = 0$ $H_{\text{máx}} = -\ln\left(\frac{1}{n}\right) = \ln\left(n\right)$
6	Entropía espectral	Calculando el espectro: $X(\omega_i)$ Energía de la densidad de espectro (PSD): $P(\omega_i) = \frac{\lvert X(\omega_i)\rvert^2}{N}$ Normalización de PSD: $p_i = \frac{P(\omega_i)}{\sum_i P(\omega_i)}$ Entropía espectral $H_S(n)$: $-\sum_{i=1}^{n} p_i \ln\left(p_i\right)$

por cero es donde las muestras tienen un cambio en los signos algebraicos. El porcentaje de los cruces por cero es una medida del contenido de la frecuencia de una señal. El promedio de los cruces por cero es un promedio razonable en la manera de estimar la frecuencia de

Tabla 2.3: Tabla de funciones para suavizar señales

Nº	*Nombre*	*Función*
1	Filtro de Pre-énfasis	$h(n) = s(n) - a * s(n-1)$
2	Medianas movíles	$m_t^{[l]} = mediana(s(t-u), \ldots, s(t), \ldots, s(t+u))$

una señal senoidal, éste promedio corresponde al valor $w(n)$. En esta ecuación $sgn(\cdot)$ es una función que regresa un valor de 1 si el valor de la $sgn(\cdot) \geq 0$, y regresa el valor de -1 en el caso contrario. $E(n)$ de la fila Nº 4 representa la energía de tiempo corto en el instante n, es decir, la suma de los cuadrados de las M muestras de la manera $n - M + 1$ a través de n. $H(n)$ de la fila Nº 5 representa la entropía cómo una medida para calcular el contenido de información de un mensaje, para ello se toma en cuenta que $v_1, v_2, \ldots, v_n$ son los posibles valores que pueden tomar las muestras. A estas muestras les corresponde las probabilidad de ocurrir $p_1, p_2, \ldots, p_n$ veces respectivamente. La información propia I de cada evento depende exclusivamente de su probabilidad p_i de ocurrir, es decir, $I(p_i)$. En esta medida se emplea el logaritmo natural (entropía de Shannon) porqué se requiere que la suma total coincida con $I(p_i)$ para el caso de dos o más eventos independientes. Entonces, la entropía H es la esperanza matemática del contenido de información en una secuencia. $H_S(n)$ de la fila Nº 6 representa la entropía espectral en la cual se puede observar que en esencia son los mismo pasos que la entropía, lo único que hace diferencia, es que se requiere del espectro de frecuencia $X(\omega_i)$ que se consigue con aplicar la Transformada Discreta de Fourier y $\omega = 2\pi f$, donde f es la frecuencia.

En la Tabla 2.3 $h(n)$ de la fila Nº 1 es el filtro de pre-énfasis el cual es un suavizado de altas frecuencias en tiempo discreto n. Éste suavizado dice que a la muestra actual ($s(n)$) se le resta la muestra anterior $s(n-1)$ suavizada por un factor de descuento a. A veces es útil superponer una versión suavizada de los datos originales en el gráfico original de las series de tiempo para ayudar a revelar patrones en los datos originales. Por éste motivo $m_t^{[l]}$ representa el suavizador de las medianas móviles que es utilizado cuando la serie de tiempo puede estar contaminada con valores de datos inusuales también conocidos cómo outliers o valores atípicos. En esta ecuación l es la duración, $l = 2u + 1$, por ejemplo, para calcular la mediana móvil en una serie de tiempo en el instante t con una duración de 3 (o también

dicho de 3 pasos) el valor que se calcula corresponde a $m_t^{[3]} = mediana(s(t-1), s(t), s(t+1))$.

2.3. Descripción de la metodología implementada

El método propuesto en este trabajo de tesis se relaciona en su mayoría al procesamiento y caracterización de señales de voz con las emociones i.e. Enojo, Aburrimiento, Disgusto, Ansiedad, Miedo, Felicidad, Tristeza, Sorpresa y Neutral. Estas emociones se eligieron porqué son una combinación de respuestas fisiológicas, comportamientos, manifiestos y sentimientos conscientes además que son emociones básicas que parecen ser persistentes en la humanidad. Aunque no todas las bases de datos emocionales están constituidas por ese conjunto de emociones, varias bases de datos comparten algunas de ellas y se complementan por otras. Estas emociones están disponibles con las bases de datos EMO-DB y SAVEE-DB en los idiomas Alemán e Inglés respectivamente (utilizar estos idiomas no significa que son los ideales para tener una representación emocional general, se utilizan porqué las bases de datos se encuentran en esos idiomas).

La Figura 2.1 muestra el proceso para la clasificación de emociones, en el cual se incluyen el pre-procesamiento que se le dio a las señales de voz. El proceso de extracción de características comienza con audios de las bases de datos los cuales son normalizados de acuerdo a:

$$s(i) = \frac{s(i)}{\sqrt{\frac{\sum_{j=0}^{M} s(j)^2}{M}}} \tag{2.1}$$

Para cada $i = 0, 1, \ldots, M$ muestra que contiene la señal s, esto permite tener valores de $[-1.0, 1.0]$. Después se aplican el filtro pasa-banda de 80 - 4000 Hz y el filtro de pre-énfasis con $a = 0.97$. Posteriormente se realiza un proceso de enmarcado o framming, lo que significa que se toman marcos de ventana con una *duración* de 30 milisegundos y un *traslape* de 10 milisegundos, es decir, la ventana tiene un *deslizamiento* de 20 milisegundos. Cada ventana $w(n)$ se obtiene de acuerdo a:

$$w(n) = s(n * deslizamiento : n * deslizamiento + duración) \tag{2.2}$$

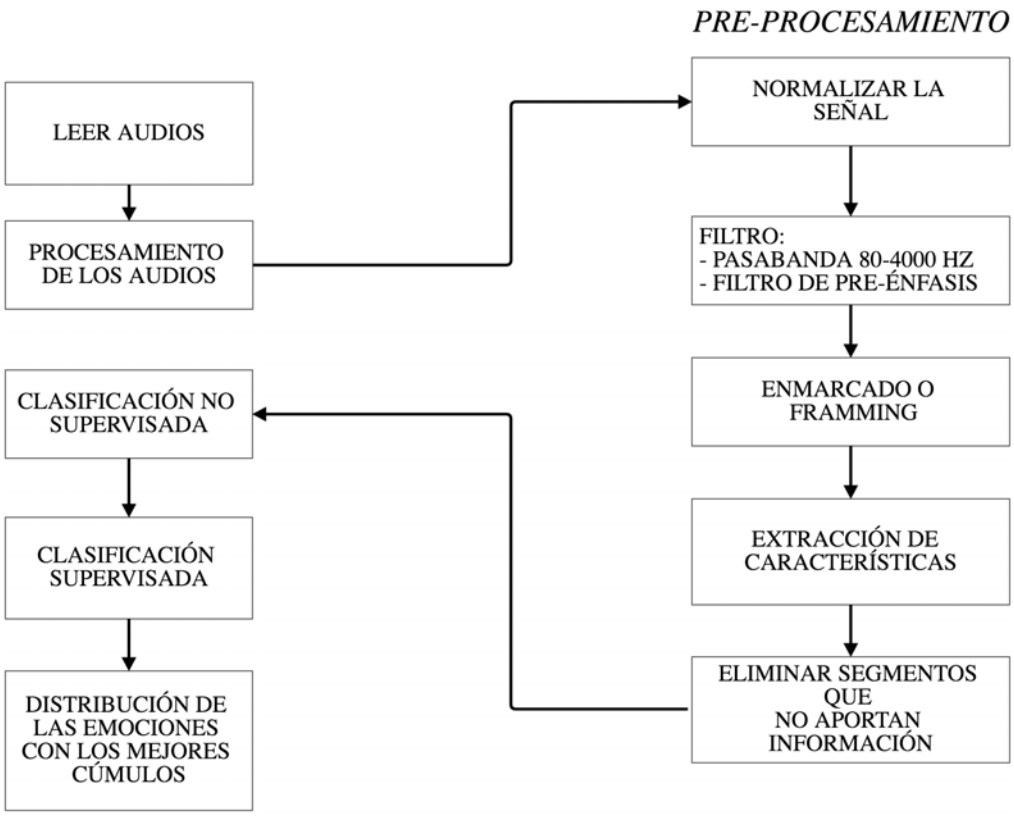


Figura 2.1: Diagrama de la clasificación implementada

Para cada $n = 0, 1, \ldots N$; dónde N es el número total de ventanas que se pueden crear.

El método propuesto para la clasificación de las emociones se compone de dos etapas. *La primera etapa* comprende la extracción de características de los audios para crear los vectores bi-dimensionales donde se guardan las mejores representaciones de las emociones. En ésta etapa se aplica al final el filtro de Medianas Móviles de los segmentos vocalizados por cada vector de características. En este trabajo de tesis se aplican los suavizados por las características de la voz antes mencionadas, además se creé que la etapa de clasificación será tan exitosa mientras que la señal este libre de ruido y otros efectos presentes cómo valores atípicos. *La segunda etapa* es la clasificación con los vectores que se obtuvieron de la etapa anterior. En ésta segunda etapa es donde se verifica que tan eficiente resulta haber procesado las señales.

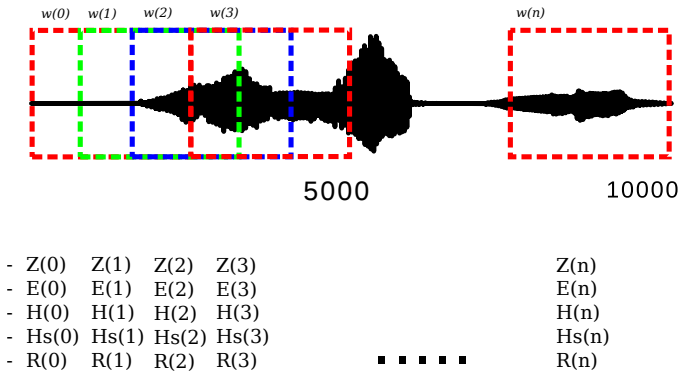


Figura 2.2: Extracción de características

En la primer etapa que se relaciona a la extracción de características. Por cada marco de ventana se obtienen cinco características las cuales corresponden i.e. Pitch, Cruces por Cero, Energía, Entropía, y Entropía Espectral. Es decir, por cada ventana el vector de características tiene la forma $X_i = \{ZCR(i), E(i), H(i), H_S(i), R(i)\}$, dónde la i hace referencia a las características que se obtienen en la i-ésima ventana. Esto se muestra con la Figura 2.2. Para obtener el Pitch se utilizaron dos maneras diferentes, las cuales son la función de ACF y la función AMDF (fila 1 y 2 de la Tabla 2.2). Los pasos que se siguieron para obtener el Pitch a partir de una señal de voz son: Primero, aplicar la $R(n)$ o $R_D(n)$ de la Tabla 2.2 para ACF ó AMDF respectivamente, por cada marco de ventana $w(n)$ de la señal de audio. Al realizar éste paso se va a obtener una señal que se puede asemejar a la Figura 2.3 ó la Figura 2.4. En estas últimas señales lo importante es detectar donde se presentan los dos principales formantes porque se va a medir la distancia entre ellos.

En las Figuras 2.3 y 2.4 se muestra con una flecha de dos direcciones la distancia entre los dos principales formantes. Una vez que se obtiene ésta distancia es dividida por la frecuencia de muestreo de la señal de voz. Cómo nota final, para encontrar los formantes se pueden utilizar diferentes estrategias, en este trabajo de tesis se optó por realizar un paso intermedio. Éste paso intermedio consiste en suavizar la señal que se obtiene por ACF o AMDF con medianas móviles de 7 pasos y obtener una señal auxiliar. A partir de las dos señales anteriores fue más fácil encontrar los formantes; la razón es que la señal que se obtiene con ACF o AMDF muestra visualmente donde se encuentran los formantes, pero

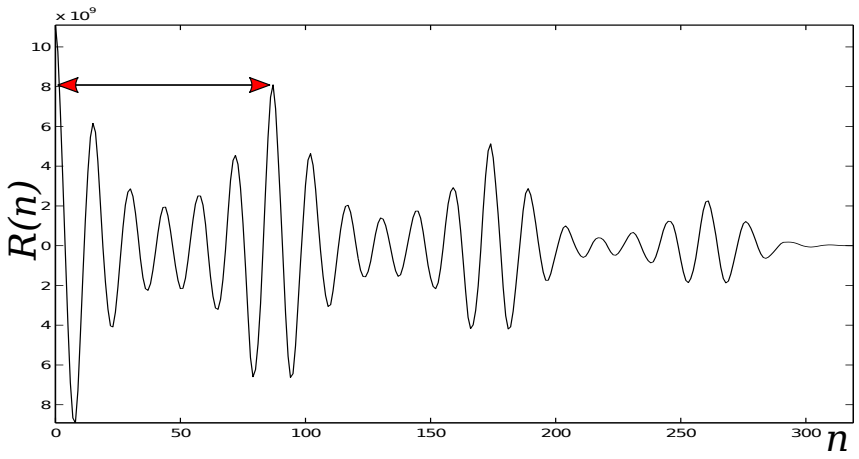


Figura 2.3: Función ACF

también está señal contiene algunos cúmulos de energía que se pueden confundir. Realizar el suavizado con las medianas móviles en este trabajo de tesis permite encontrar las mayores concentraciones de energía donde se encuentran los formantes. Se utilizarón otras estrategias para detectar el Pitch, sin embargo, no se obtienen buenos resultados por las variaciones del lenguaje y las emociones en la voz.

La Figura 2.5 muestra las características que se extrajeron de una señal de voz para la emoción de enojo. La señal superior proporciona la señal de voz completa y las siguientes cinco señales muestran las variaciones en el tiempo de cada característica que se obtuvo i.e. Cruces por cero, Energía, Entropía, Entropía Espectral y Pitch. La señal inferior muestra las zonas vocalizadas de una señal de voz.

La Figura 2.6 muestra un acercamiento de todas las variaciones de todas las características de la Figura 2.5 específicamente del primer segmento vocalizado representado por la señal de voz en dicha figura (entre líneas rojas). En este trabajo de tesis se toma en cuenta la idea de la clasificación con los filtros de Log-Gabor [Gu et al., 2016] los cuales basan su funcionamiento al análisis de señales de características cómo las que aparecen en la Figura 2.6. Sin embargo, en vez de utilizar el comportamiento completo de la señal, se va a trabajar con un estimado de las secciones que representen lo mejor posible las características emocionales de los parlantes.

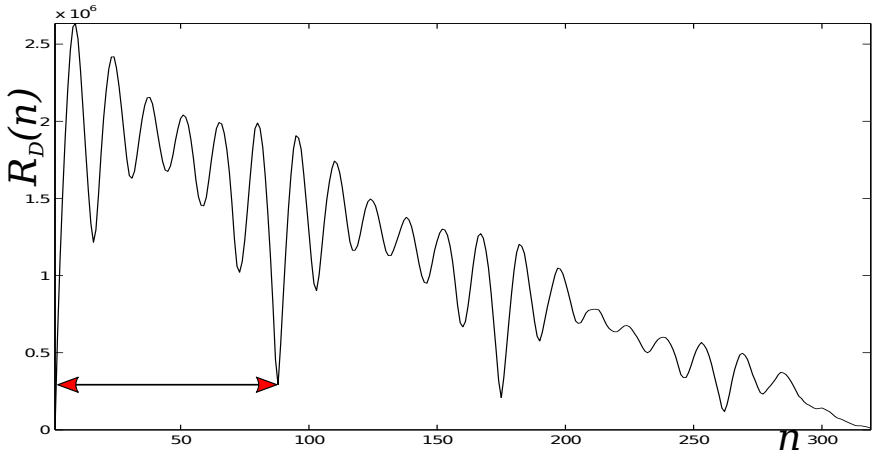


Figura 2.4: Función AMDF

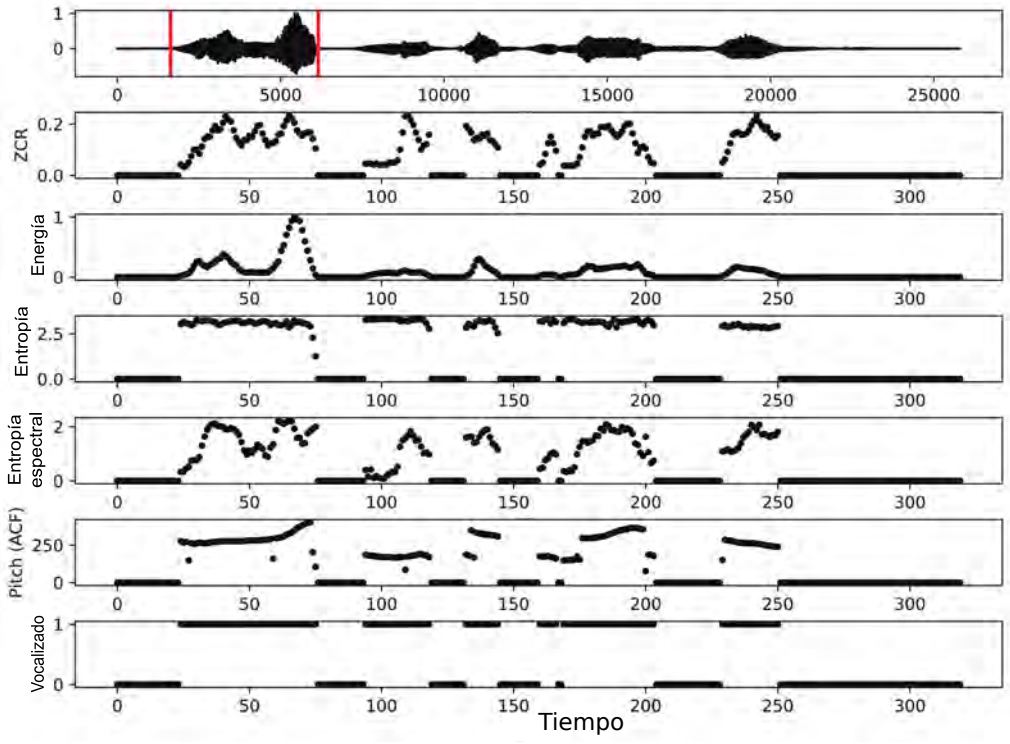


Figura 2.5: Extracción de características de una señal de voz

Hasta éste momento se puede notar que estos cinco valores por ventana se pueden ver cómo cinco nuevas señales donde su longitud depende de la cantidad total de ventanas

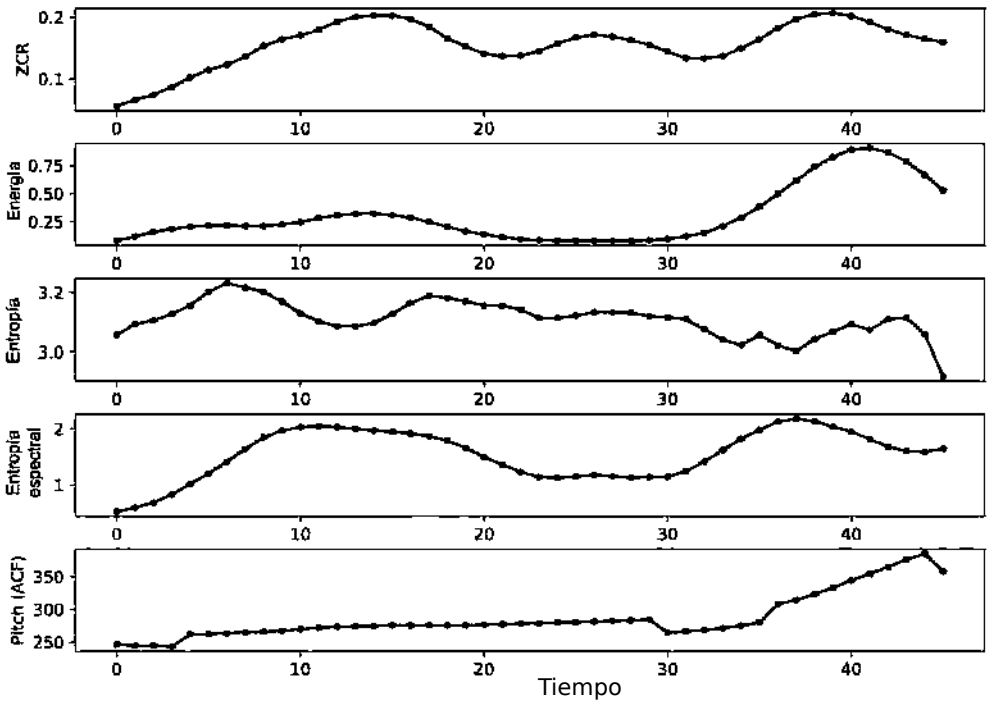


Figura 2.6: Extracción de características de un segmento vocalizado de una señal de voz

que se obtiene de cada audio original. A partir de estas señales se va solo a trabajar con los segmentos vocalizados, para poder realizar esa actividad, se va a utilizar de herramienta la señal con los valores de la energía eligiendo un valor de umbral ($th = 0.02$). Cualquier valor de la señal mayor al umbral representa un segmento vocalizado (con un valor de uno) y cualquier valor de la señal menor al umbral representa un segmento no vocalizado (con un valor de cero). Entonces, al utilizar éste valor de umbral se obtiene una sexta señal la cual se compone de unos y ceros representando los segmentos vocalizados y no vocalizados respectivamente (ver Figura 2.7). La sexta señal va ayudar para eliminar los segmentos no vocalizados que se obtienen en las señales de los Cruces por cero, Entropía, Entropía Espectral y Pitch. Después de eliminar los segmentos no vocalizados de las cinco señales anteriores, ahora es el momento de utilizar por cada segmento vocalizado de cada señal el filtro de medianas móviles para eliminar cualquier valor atípico. Es conveniente mencionar que aplicar un suavizado con medianas móviles a porciones vocalizadas muy cortas puede

fallar, por éste motivo aquellas pequeñas porciones son descartadas.

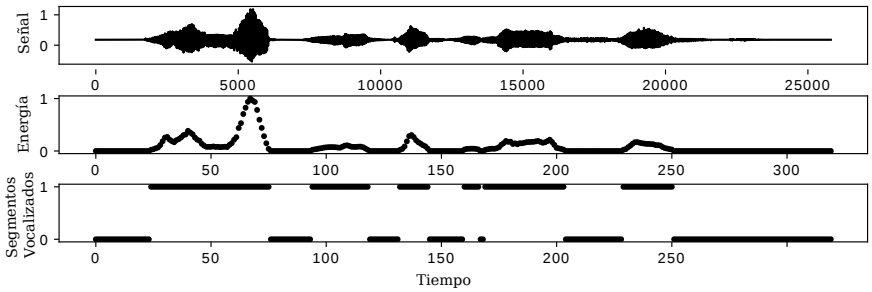


Figura 2.7: Eliminación de segmentos no vocalizados

Ahora se va a utilizar la media aritmética ($\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$) por cada segmento vocalizado y la señal completa para formar las cinco nuevas señales i.e. Cruces por ceros, Energía, Entropía, Entropía Espectral y Pitch. Se utiliza la media aritmética porqué las señales que se obtienen a partir del pre-procesamiento no experimentan cambios drásticos y de ésta manera se pueden formar acumulaciones de puntos emocionales cómo se ha hecho en la literatura.

A partir de los nuevos vectores emocionales se realiza una combinación entre ellos manteniendo el Pitch cómo principal referencia por las características antes mencionadas del Pitch en la sección anterior. Las combinaciones que se obtienen son: {Pitch, ZCR}, {Pitch, Energía}, {Pitch, Entropía}, {Pitch, Entropía espectral}, lo que significa que nos quedamos con cuatro nuevos vectores bi-dimensionales. La longitud de éstos vectores al inicio correspondía a la cantidad de ventanas creadas, posteriormente se disminuyó al descartar los segmentos no vocalizados y los segmentos vocalizados pequeños. A cada uno de estos vectores se aplica el algoritmo de k-Medias con un máximo de tres cúmulos (por las características de la voz de hombre, mujeres y niños) con la finalidad de encontrar el número de centroides que mejor representen a un estado emocional y descartar aquellos puntos que se encuentran muy alejados de los centroides utilizando k-NN (k-Vecinos más Cercanos). Cómo se puede observar en éste último proceso disminuye aun más la longitud de los vectores, preservando solo las características más representativas de las emociones. Este proceso se aplica a todos los audios contenidos en las diferentes bases de datos para obtener la o las mejores zonas donde se representan las emociones con diferentes parlantes, frases y estados

emocionales. Cómo se puede observar en estos últimos párrafos, no se agrega alguna nueva dimensión a partir de las características que se obtienen. Solamente se invierte esfuerzo en separar las características emocionales más significativas de todos los audios con los que se dispone. Para separar las características se incluye el algoritmo de k-Medias, del cual su objetivo es mover el centroide donde se representan las emocionales. A partir de ahí obtener el mejor número de puntos que representan a cada zona emocional antes de clasificar. Ésta estrategia de crear zonas emocionales y mover los centroides con el algoritmo de k-Medias es en esencia el funcionamiento de un Kernel Trick para las SVM. El Kernel Trick es definir las muestras en términos del espacio original sin conocer aun alguna función de transformación [Schölkopf, 2001, Suykens et al., 2003]. Al realizar el pre-procesamiento de la señal además del suavizado y la media aritmética de los segmentos vocalizados, lo que se esta haciendo es modificar la posición de las muestras donde se encuentran las emociones, es decir, reducir la variabilidad de cada clase emocional, que es lo que hace un kernel (transformarlas en un nuevo espacio de variables). Además en este trabajo de tesis re-construir las zonas emocionales a partir los centroides y luego clasificar se relaciona al trabajo de [Zhang et al., 2006]. En ese trabajo se propone un método híbrido con las SVM y k-NN el cual produce excelentes resultados para la clasificación.

La segunda etapa de clasificación inicia desde que se obtienen las zonas emocionales. A estas zonas se aplican varios clasificadores que se han estado utilizando en la literatura por su buena precisión. En el primer caso se encuentran las Máquinas de Soporte Vectorial (Support Vector Machines, SVM por sus siglas del inglés) [Shawe-Taylor and Cristianini, 2004], las cuales utilizan una dimensión adicional para separa las muestras con la ayuda de un hiper-plano utilizando los puntos más cercanos a las frontera de cada clase cómo vectores de soporte. En el segundo caso se encuentran los clasificadores de arboles i.e. Bosques Aleatorios (Random Forests, RF por sus siglas del inglés) [Breiman, 2001], Árboles Extra (Extra Tree, ET por sus siglas del inglés)[Geurts et al., 2006], Árboles de Decisión (Decision Trees, DT por sus siglas del inglés) [Norouzi et al., 2015] y Ada Boost [Ridgeway, 2007, Zhu et al., 2009] los cuales basan su funcionamiento para crear agrupaciones de puntos. Todos estos clasificadores son aplicados utilizando las librería Scikit-learn de Python.

En este trabajo de tesis para poder predecir el estado emocional, primero se comprueba que el redimiendo del clasificador es satisfactorio en diferentes zonas emocionales. Después se obtienen las coordenadas de los centroides de todas esas zonas con el objetivo de medir la distancia por proximidad entre las nuevas muestras emocionales a los centroides. Para poder desarrollar ésta actividad, primero se realiza el proceso de extracción de características por cada segundo de conversación con una persona diferente para obtener el vector de características de menor longitud (sin aplicar el algoritmo de k-Medias, solamente se aplica la etapa de pre-procesamiento). El siguiente paso es medir la distancia euclidiana de cada punto del vector de la nueva conversación a cada uno de los centros de las zonas emocionales que ya fueron encontradas. Y se dá por hecho donde se concentra la mayor cantidad de puntos, ese es el estado emocional que se percibe en ese momento a través de la voz.

El motivo para realizar el proceso de clasificación de ésta manera es porqué en la literatura la gran mayoría de los autores a partir de los audios extraen características muy complejas y aplican la clasificación. En muchos casos se obtienen muy buenos resultados. Sin embargo, aplicar esas ideas en un sistema de tiempo real se tardaría bastante por el hecho de que existen procesos de extracción de características con tiempo cuadráticos y cúbicos. Adicionalmente el tiempo de cómputo al clasificar es un poco costoso, sin contar el caso de que muchos autores agregan un algoritmo auxiliar para la reducción de características. En este trabajo de tesis se evita ese método y se invierte más tiempo en un procesamiento de las señales para obtener zonas donde se pueden percibir las emociones y que sean utilizables en un sistema de tiempo real.

La implementación realizada se puede resumir de la siguiente manera: Primero se aplica un *pre-procesamiento* que incluye normalizar la señal, reducción de ruido (aplicando un filtro pasa-banda de 80 - 4000 Hz y el filtro de pre-énfasis con $a = 0.97$). Entonces, se realiza un proceso de *enmarcado o framming* con un tamaño de 30 ms y traslape de 10 ms. Después de esto se aplica la *extracción de características* para cada marco de ventana, se detectan los segmentos vocalizados y se termina con la eliminación de los segmentos no vocalizados. Con esto se pasa al proceso de *eliminación de segmentos cortos* que no proporcionen suficiente información. Antes de finalizar se vuelve a suavizar los segmentos

vocalizados con medianas móviles de 11 pasos y se obtiene la media aritmética de cada segmento suavizado. Posteriormente se *re-construyen* las zonas emocionales utilizando el algoritmo de k-Medias para encontrar los mejores centroides que representan las emociones y el algoritmo k- NN para obtener a los mejores representantes. Finalmente, se aplica la *clasificación* con Bosques Aleatorios, Árboles Extra, Árboles de Decisión, Ada Boost y SVM. En la tarea de validación, el sistema utiliza un método de validación cruzada.

2.4. Clasificación del estado emocional

En esta sección se mencionan las pruebas realizadas para clasificar el estado emocional con las bases de datos EMO-DB y SAVEE-DB con un 70 % cómo conjunto de entrenamiento y un 30 % cómo conjunto de validación. Estos valores corresponden a la manera en qué se ha trabajo en la literatura para este problema de clasificación.

En la sección anterior se mencionó que a partir de las bases de datos, se extraen características para representar las emociones. Todas las características extraídas se agrupaban en vectores de dos dimensiones manteniendo el valor del Pitch cómo principal componente. Entonces, cada vector representa algún estado emocional, un ejemplo se muestra con la Figura 2.8. En ésta figura las dimensiones se componen por el par de valores {Picth, ZCR} que fueron extraídos a partir de algunos audios de la base de datos EMO-DB para la emoción de enojo. Ésta figura a simple vista parece un solo cúmulo, pero al observar con más detenimiento se muestran algunas zonas donde se comienzan a formar concentraciones de puntos. Sin embargo, también se muestran algunos otros puntos alejados lo que significa qué desde el pre-procesamiento y reducción de características aún se conservan algunos datos que pueden ser valores atípicos. Todos esos valores atípicos podrían causar un problema de clasificación cuando se tienen más emociones, porqué puede surgir el caso de que las muestras emocionales se traslapen demasiado. Omitiendo los valores atípicos de ésta figura, se puede observar que se forman dos cúmulos principales y un posible tercer cúmulo adicional.

El trabajo de esta tesis en ésta etapa de clasificación consiste en localizar el centroide por cada cúmulo emocional. A partir de cada centroide obtener el mejor número de

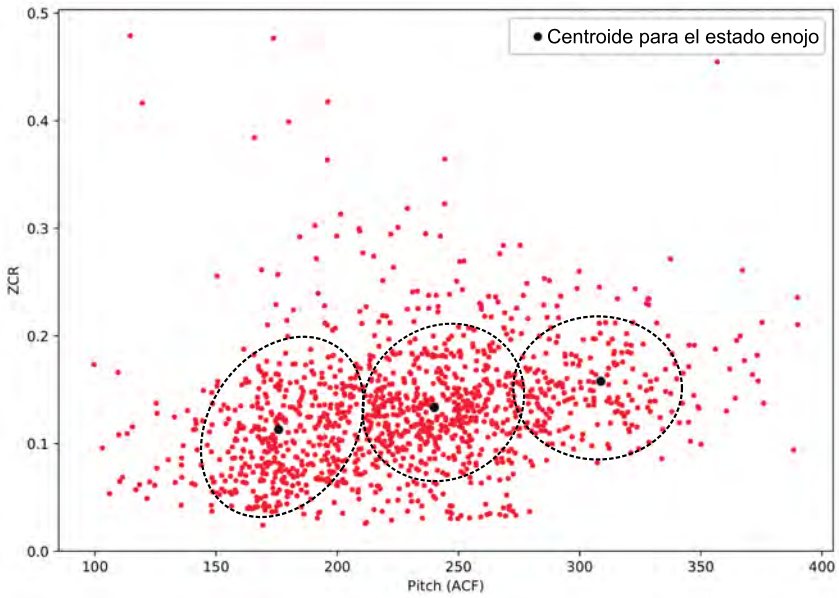


Figura 2.8: Tres cúmulos para el estado emocional enojado (EMO-DB)

puntos vecinos más cercanos para obtener alguna zona que represente lo mejor posible un estado emocional. Realizar ésta última actividad permite establecer límites entre los estados emocionales solo con datos que más influyen. Utilizar esta estrategia de zonas emocionales, permite que en una conversación al estar obteniendo muestras y extraer características si se obtienen valores que no pertenecen a las zonas ya establecidas, se pueden descartar. También por este motivo se consideran audios con duración aproximadamente de un segundo a una frecuencia de muestreo al menos de 8 kHz con ventanas de 30 msegs y translape de 10 msegs, con estas características se obtienen aproximadamente 50 puntos por segundo. Obviamente en una conversación las frases pronunciadas y turnos de los parlantes para expresarse superan un segundo, lo cual ayuda para obtener más información durante la estimación del estado emocional.

Antes de verificar el efecto de obtener varios centroides para representar cada emoción. En la Figura 2.9 se muestra cómo se distribuyen las emociones al considerar el Pitch y los ZCR con un sólo centroide por emoción quitando los valores atípicos. Ésta Figura muestra qué se tendría un problema de clasificación con las emociones de tristeza,

aburrimiento y neutral. Éste efecto tiene mucho sentido porqué las emociones de tristeza y aburrimiento tienen baja energía (por la Tabla 2.1), mientras que la emoción neutral por medio de la energía no es fácil de agruparla. De hecho en ésta Figura 2.9, se puede ver un orden descendente entre el Pitch o bien la velocidad al hablar y los ZCR. Es decir, alguna persona que experimenta enojo se expresa más rápido al igual la energía que se requiere para expresar las frases es mayor, a diferencia de una persona que experimenta tristeza, porqué la voz es más lenta y con menos energía. Lo importante de ésta figura es que las posiciones de las emociones encontradas al momento corresponden a lo que se ha estado desarrollando en la literatura, lo cual expresa que este trabajo de tesis va por buen camino.

Algunos trabajos de la literatura mencionan que la entropía es uno de los mejores parámetros para la clasificación de emociones, en la Figura 2.10 se muestra cómo se distribuyen las emociones al considerar el Pitch y la entropía con un sólo centroide por emoción quitando los valores atípicos. Lo que se puede observar a simple vista es que las emociones de tristeza, neutral y aburrimiento fueron separadas. Sin embargo, se tiene ahora un problema con las emociones de disgusto y ansiedad. Ésta Figura 2.10 es similar con la Figura 2.9 respecto a las emociones de enojo, felicidad y tristeza. Además, sucede algo similar respecto a la energía necesaria para expresar alguna emoción.

Hasta este momento con la entropía se obtiene un mejor desempeño, porqué separa las emociones de tristeza, aburrimiento y neutral de la Figura 2.9. Lamentablemente une un poco las emociones de disgusto y ansiedad. Es cuando surge la pregunta ¿Qué pasaría si en lugar de considerar una sola zona o centroide emocional cómo lo han hecho en la literatura, existieran más lugares donde se representen las emociones?. Para responder ésta pregunta la Figura 2.11 muestra que si el número de centroides del estado emocional se extiende, las muestras son más separables y en teoría debería aumentar la clasificación. Entonces, es necesario encontrar el mejor de número de centroides para representar los estados emocionales. Para resolver esta tarea se utiliza el algoritmo k-Medias donde se tiene la limitante por los sistemas vocales con un máximo de tres centroides incluyendo a hombres, mujeres y jóvenes, por cada conjunto de entrenamiento de cada estado emocional. El algoritmo k-Medias es bueno para encontrar la posición de los centroides pero cómo no se conoce exactamente cuantos cúmulos se tendrán, es entonces cuando se utiliza un algoritmo

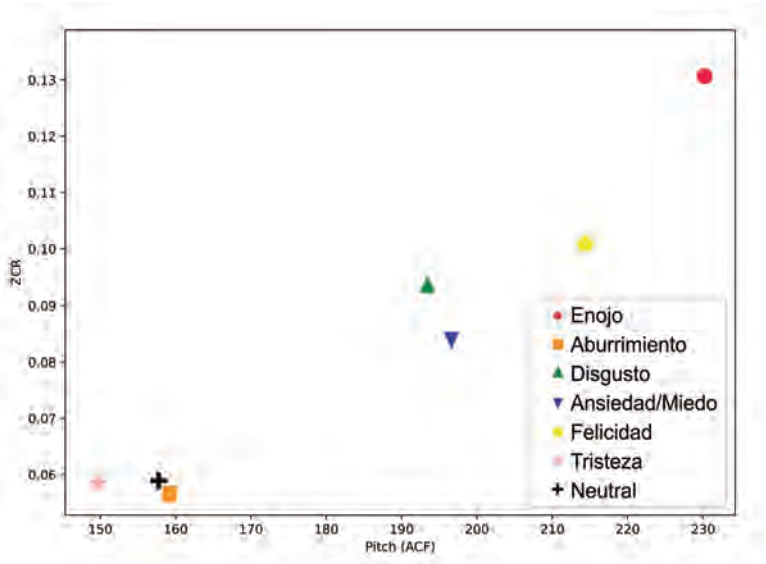


Figura 2.9: Un centroide para cada emoción con Pitch vs ZCR (EMO-DB)

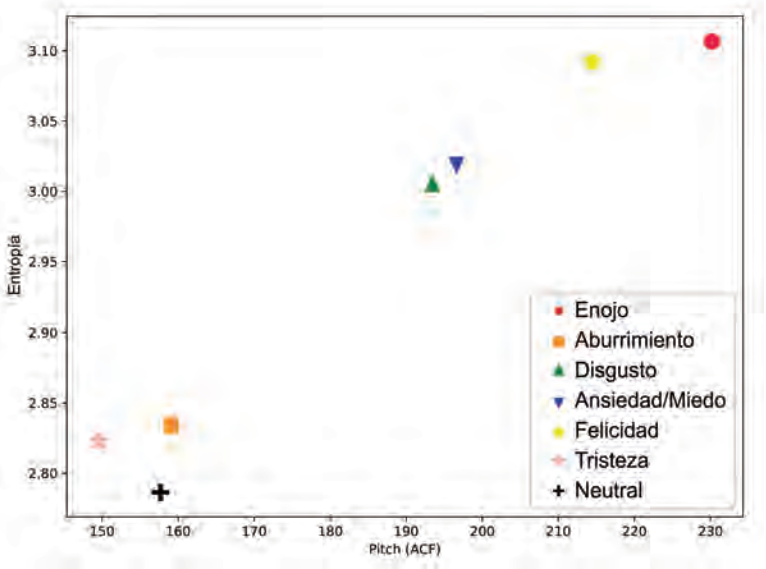


Figura 2.10: Un centroide para cada emoción con Pitch vs Entropía (EMO-DB)

iterativo que se encarga de buscar el mejor número de centroides y en cada búsqueda se realiza una clasificación. Donde se obtiene la mejor precisión, ahí se encuentra el valor de la k.

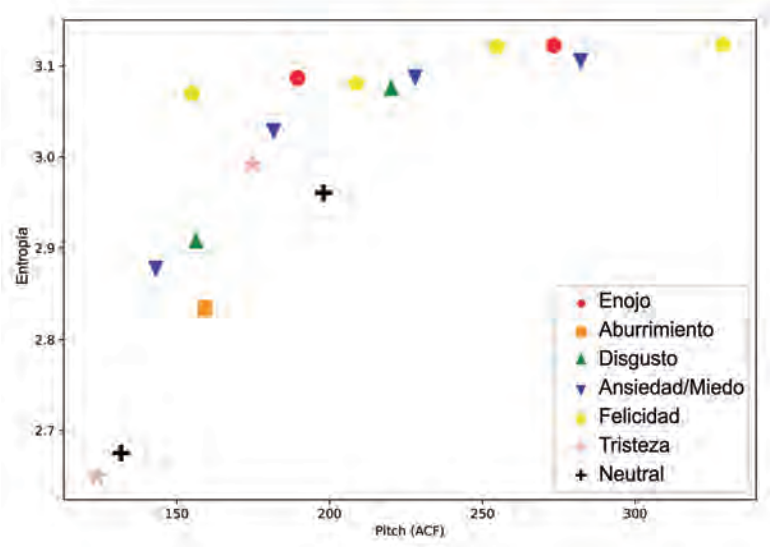


Figura 2.11: Varios centroides para las emociones con Pitch vs Entropía (EMO-DB)

Entonces, en los experimentos realizados se toman en consideración dos escenarios para la clasificación: El primero con un solo centroide y el segundo con varios centroides por emociones. Durante el primer escenario al utilizar un solo centroide para representar las emociones se obtienen resultados pobres (es decir, muy bajos) en comparación con la literatura pero sólo en el caso del clasificador SVM y Ada Boost, esto se debe a su manera de clasificar porqué no se tiene un kernel adecuado para éste tipo de muestras. Las Tablas 2.4 y 2.5 muestran la precisión que se obtuvó con un solo centroide para las bases de datos EMO-DB y SAVEE-DB. En éstas pruebas se utiliza el Pitch y cada uno de los pares de elementos de la primera columna. En estas tablas se puede observar que la clasificación con DT y ET obtuvó la mejor precisión seguido de RF, Ada Boost y finalmente SVM con la precisión más pobre.

Tabla 2.4: Clasificación con un centroide por emoción (EMO-DB)

	SVM Linear %	*Decision Trees* %	*Random Forest* %	*Extra Trees* %	*Ada Boost* %
{Pitch,ZCR}	**68.97**	**99.9**	**99.9**	**99.9**	88.97
{Pitch,Energía}	68.16	**99.9**	**99.9**	**99.9**	90.20
{Pitch,Entropía}	67.34	**99.9**	**99.9**	**99.9**	**94.69**
{Pitch,Entropía espectral}	68.16	**99.9**	99.59	**99.9**	91.02

Tabla 2.5: Clasificación con un centroide por emoción (SAVEE-DB)

	SVM Linear %	*Decision Trees %*	*Random Forest %*	*Extra Trees %*	*Ada Boost %*
{Pitch,ZCR}	35.88	**99.9**	**99.9**	**99.9**	**71.77**
{Pitch,Energía}	35.48	**99.9**	**99.9**	**99.9**	61.69
{Pitch,Entropía}	**37.90**	**99.9**	**99.9**	**99.9**	60.88
{Pitch,Entropía espectral}	35.88	**99.9**	99.4	**99.9**	66.93

En el segundo escenario se encontró que el mejor número de centroides para el clasificador SVM con kernel lineal, y la base de datos EMO-DB es [1, 2, 1, 3, 2, 1, 3] que corresponde a 1 centroide para enojo, 2 centroides para aburrimiento, 1 centroide para disgusto, 3 centroides para ansiedad/miedo, 2 centroides para felicidad, 1 centroide para tristeza y finalmente 3 centroides para neutral respectivamente. Para la base de datos SAVEE-DB es [2, 1, 1, 2, 2, 3, 3] que corresponde a 2 centroides para enojo, 1 centroide para disgusto, 1 centroide para miedo, 2 centroides para felicidad, 2 centroides para neutral, 3 centroides para tristeza y 3 centroides para sorpresa respectivamente.

Tabla 2.6: Clasificación con [1, 2, 1, 3, 2, 1, 3] centroides por emoción (EMO-DB)

	SVM Linear %	*Decision Trees %*	*Random Forest %*	*Extra Trees %*	*Ada Boost %*
{Pitch,ZCR}	91.70	**99.9**	**99.9**	**99.9**	**99.16**
{Pitch,Energía}	91.28	**99.9**	**99.9**	**99.9**	98.76
{Pitch,Entropía}	91.28	**99.9**	**99.9**	**99.9**	97.5
{Pitch,Entropía espectral}	**92.11**	**99.9**	**99.9**	**99.9**	98.75

Tabla 2.7: Clasificación con [2, 1, 1, 2, 2, 3, 3] centroides por emoción (SAVEE-DB)

	SVM Linear %	*Decision Trees %*	*Random Forest %*	*Extra Trees %*	*Ada Boost %*
{Pitch,ZCR}	**84.08**	**99.9**	**99.9**	**99.9**	94.28
{Pitch,Energía}	74.89	**99.9**	**99.9**	**99.9**	94.69
{Pitch,Entropía}	83.67	**99.9**	**99.9**	**99.9**	**97.55**
{Pitch,Entropía espectral}	**84.08**	**99.9**	**99.9**	**99.9**	94.71

Las Tablas 2.6 y 2.7 muestran la precisión que se obtuvo con un número diferente de centroides. En estas tablas se puede obervar que la peor presición se obtiene nuevamente por las SVM. El clasificador SVM ha sido el último lugar en los experimentos previos. Una pregunta que surge de los resultados anteriores es: ¿Cuáles son las emociones que afectan al rendimiento del clasificador SVM?. Las Tablas 2.8 y 2.9 pueden ser una respuesta, en estas

tablas se muestra las precisión por estado emocional cambiando cada kernel por el elemento en la primera columna. En general, se puede observar que las emociones de enojo con tristeza para EMO-DB y disgusto con miedo para SAVEE-DB obtienen la mejor precisión. En ambos casos el kernel lineal sobresale a diferencia del kernel RBF. En la Tabla 2.9 no se agregó un kernel polinomial porqué, además de que el tiempo aumenta al clasificar el conjunto de datos que se obtiene, se pierde precisión al aumentar el grado del polinomio.

Tabla 2.8: Precisión promedio de múltiples kernels para SVM (EMO-DB)

Kernel	*Enojo %*	*Aburrimiento %*	*Disgusto %*	*Ansiedad %*	*Felicidad %*	*Tristeza %*	*Neutral %*	*Promedio*
Lineal	100	87.5	90.90	86.11	96.42	100	78.53	**91.35**
RBF	100	41.66	100	44.44	75.93	100	59.68	74.53
Poli 3	100	73.33	100	74.74	93.79	100	77.46	88.47

Tabla 2.9: Precisión promedio de múltiples kernels para SVM (SAVEE-DB)

Kernel	*Enojo %*	*Aburrimiento %*	*Disgusto %*	*Ansiedad %*	*Felicidad %*	*Tristeza %*	*Neutral %*	*Promedio*
Lineal	58.65	100	93.10	79.60	91.99	100	66.66	**84.28**
RBF	51.13	100	93.10	44.07	100	54.04	62.96	72.18

Las Tablas 2.8 y 2.9 también muestran que el reconocimiento de las emociones a través de la voz en SI obtiene una mayor precisión que los enfoques anteriores. Cómo mejores características resalta la entropía, la cual es un parámetro que se encarga de medir la cantidad de información contenida en la señal. Se podría pensar que durante la tarea del reconocimiento de emociones la energía influye más porqué, se ha mencionado que para expresar las emociones se requiere de cierta cantidad de energía. Sin embargo, toda esa información se puede apreciar de una mejor manera con el uso de la entropía y entropía espectral. En general, se pueden hacer cuatro observaciones. En primer lugar, la precisión de clasificación mostrada en las tablas de uno a varios centroides de emociones para los clasificadores SVM y Ada Boost. Estas tablas indican un aumento en precisión con emociones distribuidas en SI. En segundo lugar, los DT, RF y ET tienen la mayor precisión con 99-100 % en SI. Esto significa, que para éste tipo de problemas los clasificadores basados en árboles se comportan muy bien ya que se basan en cúmulos de datos y aunque el SVM tiene un fuerte soporte matemático, en éste tipo de problemas donde los datos se encuentran muy traslapados tiene un rendimiento pobre y para mejorarlo se necesitaría encontrar un kernel que se adapte mejor para separar los datos. En tercer lugar, se dijo anteriormente

Tabla 2.10: Precisión promedio por emoción de múltiples características con SVM-Linear (EMO-DB)

	Enojo %	*Aburrimiento %*	*Disgusto %*	*Ansiedad %*	*Felicidad %*	*Tristeza %*	*Neutral %*	*Promedio*
{Pitch,ZCR}	100	87.5	90.90	83.33	96.42	100	78.53	90.95
{Pitch,Energía}	100	87.5	90.90	83.33	92.85	100	78.53	90.44
{Pitch,Entropía}	100	83.33	90.90	83.33	96.42	100	78.53	90.36
{Pitch,Entropía espectral}	100	87.5	90.90	86.11	96.42	100	78.53	**91.35**

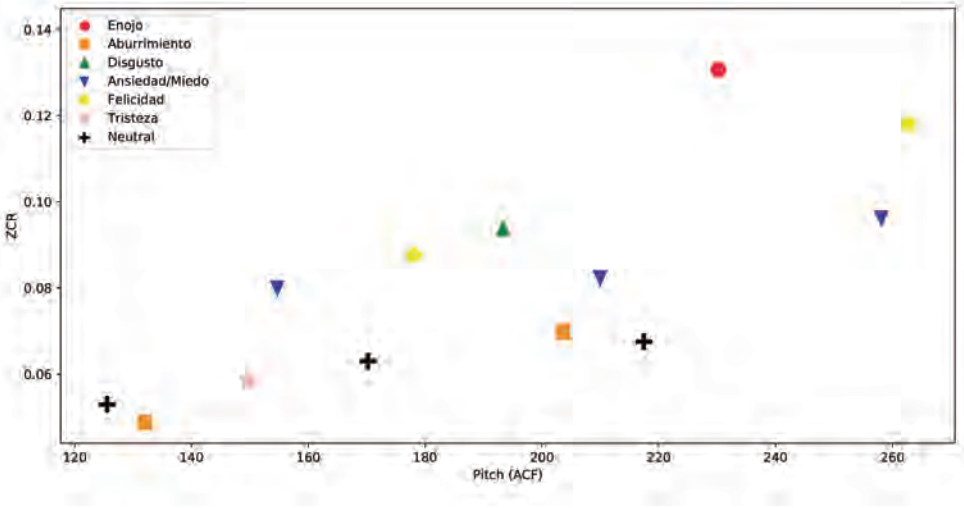
Tabla 2.11: Precisión promedio por emoción de múltiples características con SVM-Linear (SAVEE-DB)

	Enojo %	*Aburrimiento %*	*Disgusto %*	*Ansiedad %*	*Felicidad %*	*Tristeza %*	*Neutral %*	*Promedio*
{Pitch,ZCR}	58.65	100	93.10	79.60	91.99	100	66.66	**84.28**
{Pitch,Energía}	65.14	100	89.65	46.05	87.11	79.79	58.79	75.22
{Pitch,Entropía}	57.91	100	93.10	82.23	87.11	100	66.66	83.86
{Pitch,Entropía espectral}	58.65	100	93.10	79.60	91.99	100	66.66	**84.28**

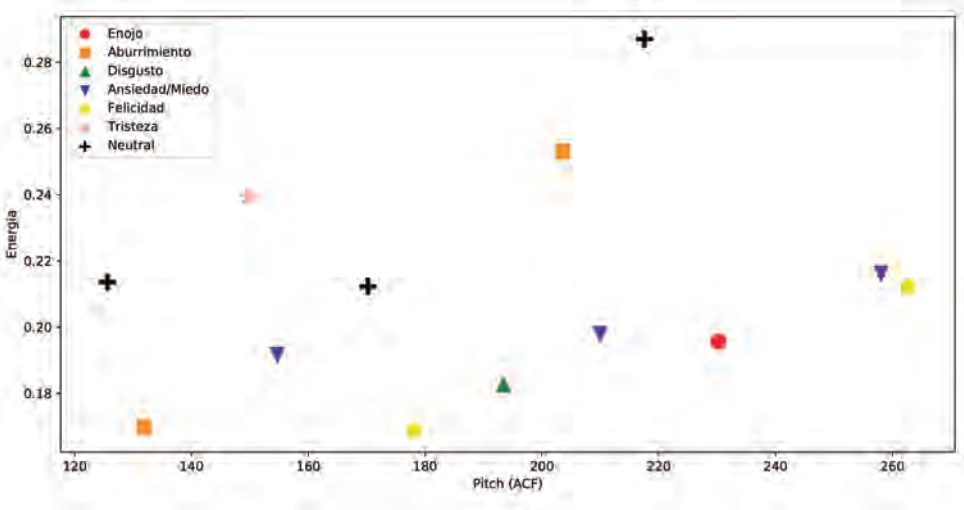
que las características también determinan el tipo de clasificador. Esto se muestra en las Tabla 2.8 y 2.9 donde la peor precisión fue para el kernel RBF para SVM con el mejor cúmulo encontrado por un SVM con kernel lineal. Finalmente, el costo computacional no es demasiado alto en la etapa de entrenamiento de los clasificadores DT, RF, ET y Ada Boost. Sin embargo, para SVM es más costoso, pero para la validación es en general rápida.

En la literatura es más común encontrar la clasificación del estado emocional con las señales de voz y las SVM cómo se muestra con las Tablas 2.10 y 2.11. En éstas tablas el valor del Pitch es común con cualquier características de la primer columna, en la última columna se muestra el promedio de la clasificación y en las otras columnas la precisión que se obtiene al clasificar una emoción contra el resto. En estos tipos de tablas es donde los autores comentan acerca de las emociones que son más fáciles de clasificar, un ejemplo es el trabajo de Poria [Poria et al., 2016a] el cual muestra que la tristeza y el enojo se detectan más fácilmente, a diferencia de ansiedad/miedo con la base de datos EMO-DB.

Las Figuras 2.12 y 2.13 para EMO-DB y las Figuras 2.14 y 2.15 para SAVEE-DB muestran la distribución emocional con los diferentes números de centroides encontrados para las emociones. Cada una de estas figuras están agrupadas por pares i.e. {Pitch, ZCR}, {Pitch, Energía}, {Pitch, Entropía}, {Pitch, Entropía espectral}. La precisión que se obtuvó en la clasificación se debe especialmente a estas distribuciones por zonas emocionales. Este enfoque es diferente respecto a varios trabajos que existen la literatura, donde ellos utilizan solamente un centroide o una zona para la distribución emocional.

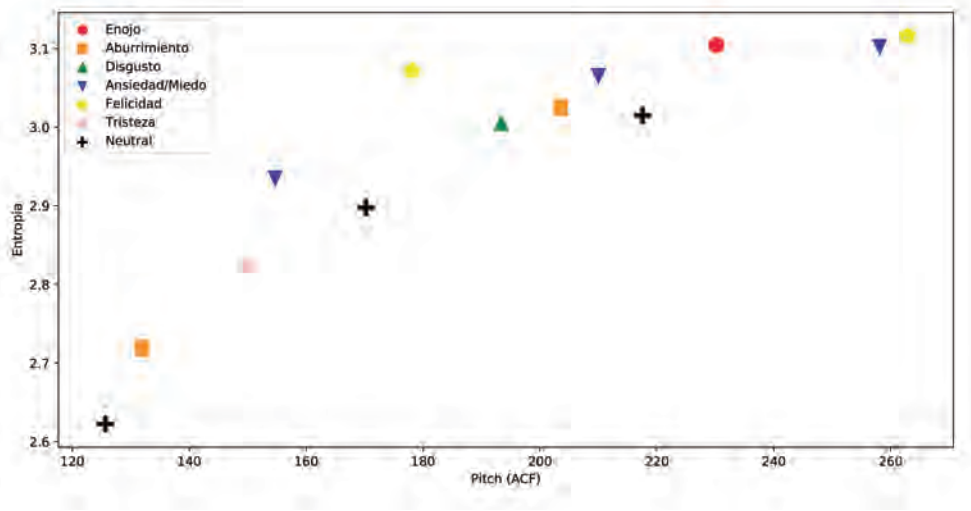


(a) ***Pitch vs ZCR***

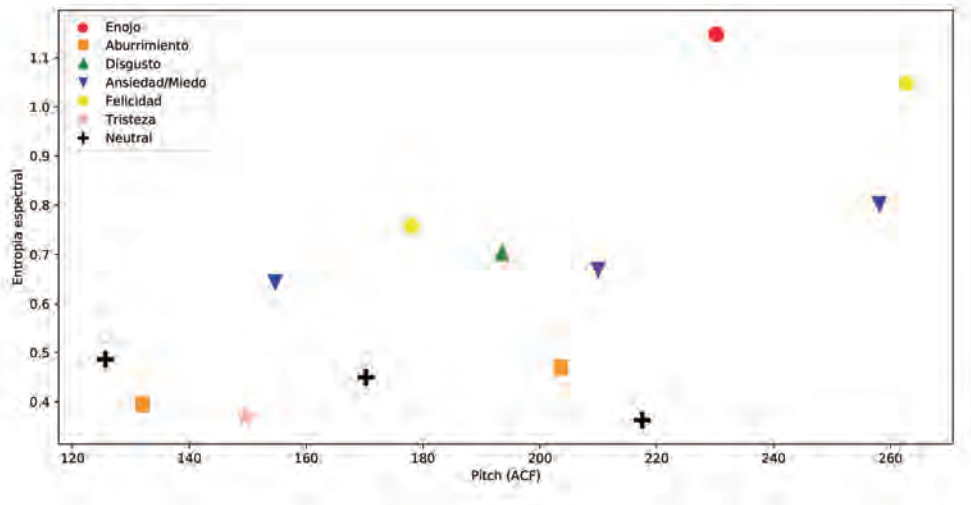


(b) ***Pitch vs Energía***

Figura 2.12: Distribución emocional con las mejores características (EMO-DB) (1/2)

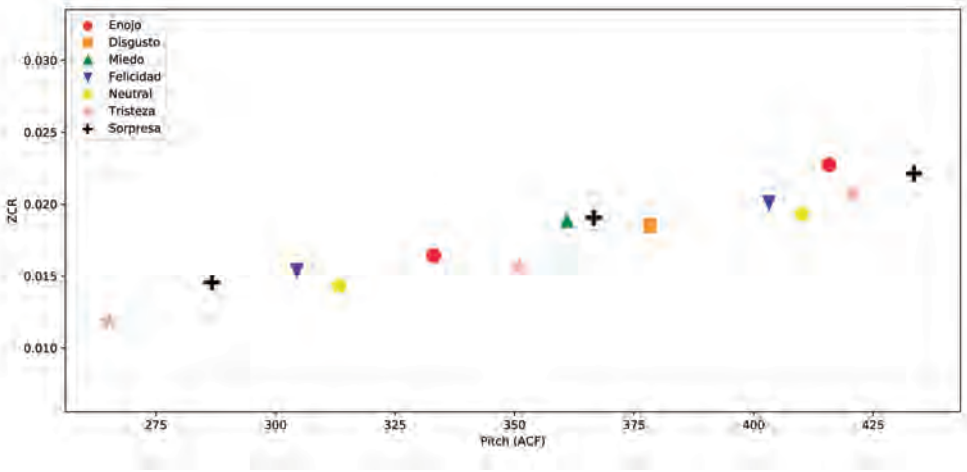


(a) ***Pitch vs Entropía***

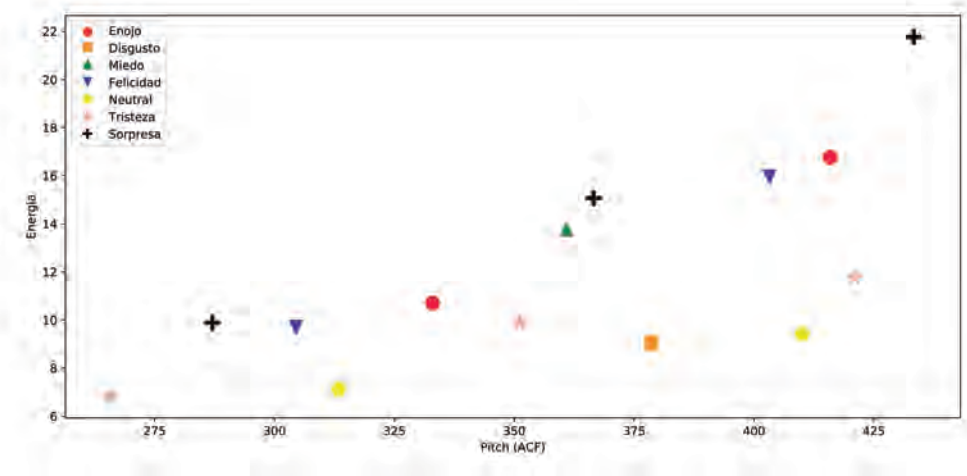


(b) ***Pitch vs Entropía Espectral***

Figura 2.13: Distribución emocional con las mejores características (EMO-DB) (2/2)

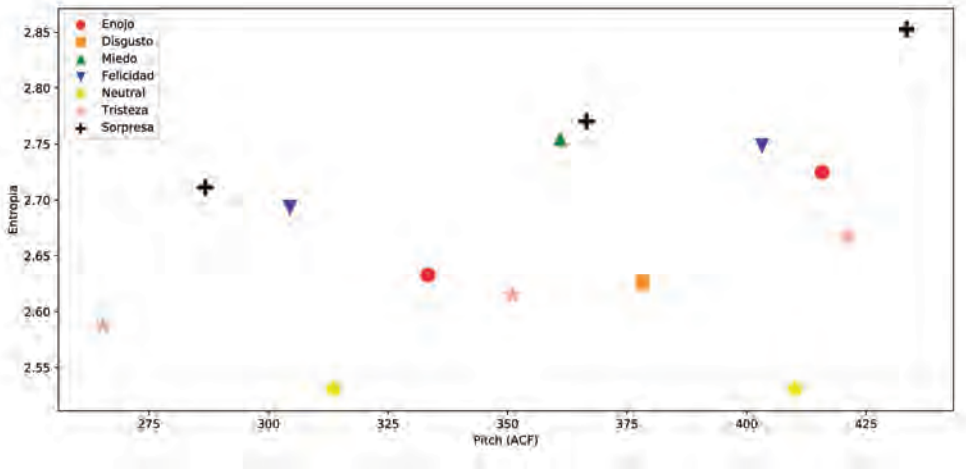


(a) ***Pitch vs ZCR***

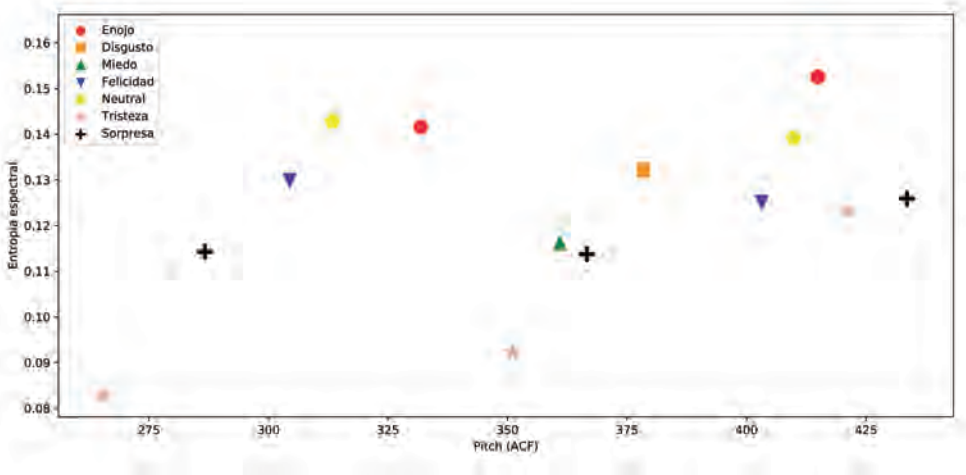


(b) ***Pitch vs Energía***

Figura 2.14: Distribución emocional con las mejores características (SAVEE-DB) (1/2)



(a) ***Pitch vs Entropía***



(b) ***Pitch vs Entropía Espectral***

Figura 2.15: Distribución emocional con las mejores características (SAVEE-DB) (2/2)

El desempeño de los algoritmos implementados se puede obtener mediante un procedimiento que consiste en medir la distancia entre un vector de características $\mathbf{X}$ y la media de distribución hacia alguna clase. Esto es interpretado de acuerdo a:

- Si la distancia es inferior a un valor de umbral *th* y el vector pertenece a dicha clase, se dice que es un *positivo-verdadero* (TP).
- Si la distancia es superior a *th* pero el patrón sí pertenece a la clase, entonces se trata de un *negativo-falso* (FN).
- Si la distancia es inferior a *th* pero el patrón en realidad no pertenece a dicha clase, esto corresponde a un *positivo-falso* (FP).
- Si la distancia es superior a *th* y el patrón no pertenece a la clase, se dice que es un *negativo-verdadero* (TN).

El régimen de precisión se define cómo la fracción de patrones correctamente reconocidos (positivos verdaderos) entre el número de consultas ejecutadas (positivos verdaderos mas positivos falsos). El *Régimen de Positivos verdaderos* (TPR) o también conocido cómo *recall* es la fracción que el sistema reconoce correctamente (positivos verdaderos) respecto al total de los que debían de haber sido reconocidos (positivos), esto se muestra en la Ecuación 2.3. El *Régimen de Positivos Falsos* (FPR) o *falsas alarmas* es una medida de frecuencia con la que el clasificador se equivoca, esto se define mediante la Ecuación 2.4. La curva ROC es un espacio característico de operación del receptor, donde el eje vertical es el TPR y el eje horizontal es el FPR. Un punto en éste plano representa el desempeño del clasificador para un cierto valor de umbral *th*, el cual genera la curva ROC cuando éste valor de umbral *th* se va modificando.

$$TPR = \frac{TP}{TP + FN} \tag{2.3}$$

$$FPR = \frac{FP}{FP + TN} \tag{2.4}$$

Las curvas ROC con la clasificación de las SVM correspondientes a las bases de datos EMO-DB y SAVEE-DB, se muestran en la Figura 2.16. En ésta figuras se puede

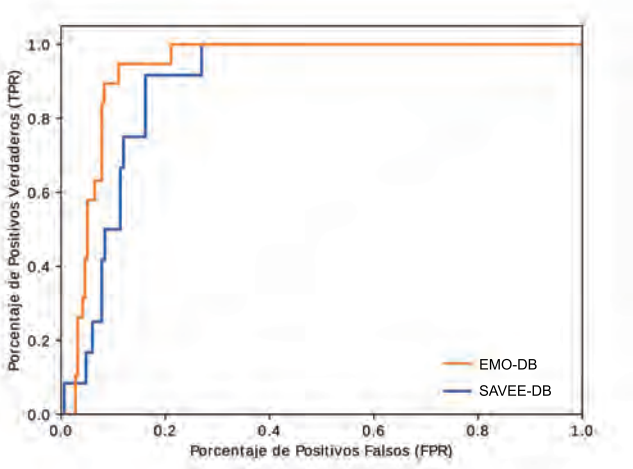


Figura 2.16: Curvas ROC de SVM con kernel lineal

observar con el área bajo la curva de los clasificadores que tienen un buen rendimiento. Ésta característica se determina por el crecimiento de la curva cómo se aleja de la diagonal, la cual corresponde a un mal clasificador.

Las Tablas 2.12 y 2.13 para las bases de datos SAVEE-DB y EMO-DB respectivamente, muestran varias características y clasificadores para SER en SI reportados por varios autores. En estas tablas se incluye el método propuesto de este trabajo de tesis. Las características extraídas se muestran en la columna 2. La tercera columna muestra los métodos de clasificación utilizados. La última columna muestra la precisión de cada una de las características utilizadas. Estas tablas describen que el método propuesto supera a varios trabajos de la literatura con características fáciles de extraer y con métodos de clasificaciones comunes cómo el caso de las Máquinas de Soporte Vectorial y los Bosques Aleatorios, además de otros nuevos cómo son los Arboles de Decisión, Arboles Extra y Ada Boost. Estas tablas también muestran que el enfoque de los autores se relaciona a extraer una amplia cantidad de características que son relacionadas especialmente a la energía, frecuencia y cantidad de información con algoritmos costosos. Para finalizar, cabe resaltar que el método propuesto no utiliza una etapa de selección de características, en su lugar se utiliza otro método que evitó el problema de la alta dimensión.

Tabla 2.12: Resumen de características y métodos de clasificación SI (SAVEE-DB)

	Características	*Método de clasificación*	*Precisión promedio %*
Yaacob et al. [Yaacob et al., 2015]	Características de entropía y wavelets con energía	Kernel ELM	77.92
Yaacob et al. [Yaacob and Polat, 2017]	Características Biespectrales	ELM	73.81
Yogesh et al. [Yogesh et al., 2017]	Formas de onda glotal y señales glotales con algoritmos de selección de características	PSOBBO	62.50
Badshah et al. [Badshah et al., 2016]	MFCC	Árboles aleatorio y Árboles de decisión	66.5 - 82.21
Método propuesto Multi-cúmulo	{Pitch,ZCR} y {Pitch,Entropía espectral}	SVM	84.08
	{Pitch,ZCR}, {Pitch,Energía}, {Pitch,Entropía} {Pitch,Entropía espectral}	Árboles de Decisión Aleatorios y Extra	99.9
	{Pitch,Entropía}	Ada Boost	94.71

Tabla 2.13: Resumen de características y métodos de clasificación SI (EMO-DB)

	Características	*Método de clasificación*	*Precisión promedio %*
Ayadi et al. [El Ayadi et al., 2011]	Características continuas, cualitativas, espectrales y características TEO, con algoritmos de selección de características	HMM GMM ANN SVM	75.5 -78.5 74.83 - 81.94 51.19 - 70 75.45 - 81.29
Rybka and Janicki et al. [Rybka and Janicki, 2013]	Coeficientes MFCC, LPC y LFPC Características basadas en F0, Energía, velocidad de la voz y parámetros de calidad de la voz con algoritmos de selección de características	k-NN ANN SVM	63.22 56 69-72
Ramakrishnan and Emaryin et al. [Ramakrishnan and El Emary, 2013]	Características acústicas y lingüísticas con algoritmos de selección de características	HMM SVM	77.6 79.7
Yaacob et al. [Yaacob et al., 2015]	Formas de onda glotal, Energía y Entropía con algoritmos de selección de características	ELM kNN	97.24 93.90
Yogesh et al. [Yogesh et al., 2017]	Formas de onda glotal y señales glotales con algoritmos de selección de características	PSOBBO	90.31
Zhang et al. [Zhang et al., 2017]	MFCC, Pitch, y Formantes con algoritmos de selección de características	SVM-Multiclass DBN	84.54 94.6
Método propuesto Multi-cúmulo	{Pitch,Entropía espectral}	SVM	92.11
	{Pitch,ZCR}, {Pitch,Energía}, {Pitch,Entropía} {Pitch,Entropía espectral}	Arboles de Decisión Aleatorios y Extra	99.9
	{Pitch,ZCR}	Ada Boost	99.16

2.5. Comentarios finales

En este capítulo se expusieron algunos modelos emocionales y características que son necesarios para realizar una clasificación del estado emocional más prometedora. Al igual se muestran algunos clasificadores y una descripción de las bases de datos emocionales que han sido más utilizados por varios investigadores. Estas ideas sirven de referencia para tratar las señales de voz y señales con las características que se extraen cómo series de tiempo. A partir de estas series de tiempo se obtienen algunos parámetros para representarlas y de está

manera crear los cúmulos naturales del espacio de características bi-dimensional. En este capítulo también se muestra la precisión promedio que se obtuvo en la etapa de clasificación. Ésta precisión compite con el estado del arte para el área del reconocimiento de emociones a través de la voz. En esta parte de la clasificación se realizaron dos observaciones principales i.e. (1) Tener varias representaciones emocionales le ayudan a las Máquinas de Soporte Vectorial para mejorar en clasificación con el kernel lineal. (2) Durante la clasificación la mejor precisión en general se relaciona al Pitch con la entropía y entropía espectral. En el siguiente capítulo se describen los algoritmos enfocados para tomar decisiones a corto-mediano plazo los cuales van ayudar para la predicción del estado emocional.

Capítulo 3

Estimación del estado emocional

En el capítulo anterior se explicó acerca de la clasificación del estado emocional con señales de voz. Resolver está tarea ayuda a determinar qué estado emocional de los parlantes se detecta en cada tiempo. Sin embargo, en una conversación se piensa que los parlantes experimentan multi-emociones, es decir, más de una emoción a la vez. En este capítulo se va a mencionar una manera para monitorear los estados emocionales en una conversación, la cual permite realizar un seguimiento y predicción del estado emocional unos pasos adelante. El seguimiento se refiere a observar en cualquier instante de tiempo el comportamiento de los niveles emocionales del parlante durante alguna conversación. La finalidad de éste capítulo es mostrar de una manera sencilla los Procesos de Decisión Parcialmente Observables de Markov.

3.1. Introducción

En la actualidad los Procesos de Decisión Parcialmente Observables de Markov mejor conocidos como POMDP (más adelante se definen formalmente) han sido utilizados en diferentes áreas, la más común es el área de la robótica móvil [Hoey et al., 2016]. Otro ejemplo es el área de control, donde se trabaja con muestras a las que se les agrega ruido. En ésta última los autores Zhou, Fu y Marcus et al. [Zhou et al., 2010] utilizan los POMDP y una proyección con el filtro de partículas para trabajar con éste tipo de muestras. Los POMDP han sido adaptados a otros sistemas de interacción con las perso-

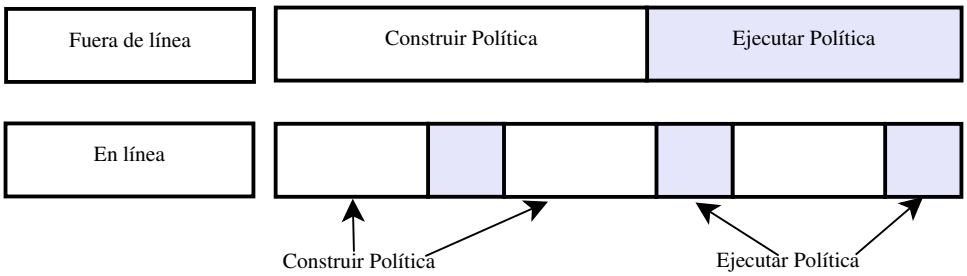


Figura 3.1: Tipos de políticas

nas, en los cuales se busca obtener una respuesta rápida y precisa. Con éste precedente surgen los Sistemas de Diálogo Hablado (Spoken Dialog Systems, SDS por sus siglas del inglés) [Williams and Young, 2007, Young et al., 2010]. Los autores Young, Gašić, Keizer et al. [Young et al., 2010] utilizan los SDS-POMDP para la interacción con el usuario especialmente en centros de llamadas inteligentes porque estos sistemas cada vez muestran una interacción más natural y de ésta manera se reduce la cantidad de costos humanos.

El último enfoque más común es la combinación de los POMDP con la optimización de Monte Carlo. Ésta optimización ha sido extensamente estudiada donde los autores Silver, David y Veness et al. [Silver and Veness, 2010] la utilizan para resolver tres problemas extensos. Estos problemas son 'Rocksample', 'Battleship' y 'Pocman' con la cantidad de estados $12'544$, 10^{18} y 10^{56} [Kyriakos,] respectivamente, además de una planificación larga con un horizonte $>$ 100,000 pasos adelante. Por otro lado los autores Chong, Kreucher y Hero et al. [Chong et al., 2008] han utilizado ésta optimización Monte-Carlo en problemas que involucra sensores para el seguimiento y detección de objetos. Los autores Thrun, Sebastian et al. [Thrun, 2000] utilizan la optimización Monte-Carlo además de combinar el algoritmo de los vecinos cercanos (k-NN) para el seguimiento y detección de la posición de los robots.

Los trabajos mencionados anteriormente son aplicaciones de los POMDP, cómo se puede observar la mayoría de las aplicaciones son hechas para robots. Sin embargo, en [Hoey et al., 2016] se presenta un proyecto titulado cómo 'BayesAct' (más adelante se define formalmente) el cual se ha estado trabajando con los POMDP sobre el estado emocional. En éste trabajo el funcionamiento de los procesos de decisión se aplican al evaluar diferentes usuarios de diferentes culturas y cambios emocionales. El funcionamiento de éste trabajo

es interesante por su rendimiento debido a que en Poria [Poria et al., 2016b] se menciona que la fase de predicción del estado emocional requiere la comprensión del contexto. Por lo tanto, esto lo convierte en una tarea compleja respecto a la clasificación de emociones.

El funcionamiento de los procesos de decisión que se han mencionado requieren de políticas. Una política es una actividad que se relaciona a la toma de decisiones la cual puede ser inmediata o a largo plazo (más adelante se define formalmente una política). Las políticas pueden ser *Fuera de linea* y *En linea*. La Figura 3.1 muestra estos dos escenarios, en ésta figura se puede observar el escenario de las políticas fuera de linea, en el cual, primero se invierte tiempo en construir las políticas y posteriormente se ejecutan. El otro escenario son las políticas en linea, en el cual se invierte solo el tiempo necesario en construir y ejecutar la política mientras el agente ésta obteniendo información.

Finalmente, durante el proceso de toma de decisiones se requiere de algún algoritmo que se encargue de tomar una decisión inmediata o a largo plazo. En éste tipo de problemas, tomar una decisión se puede realizar de manera *Exacta* y *Aproximada*, es decir, se puede seguir un proceso de actualización en base al Teorema de Bayes o se puede utilizar una estimación de la toma de decisiones pasadas. Todos éstos conceptos se van a estar detallando a lo largo de éste Capítulo.

3.2. Algoritmos para toma de decisiones

En esta sección se van a presentar los algoritmos que existen en el estado del arte para la toma de decisiones: Primero, se van a introducir los Procesos de Decisión de Markov, en los cuales se explican los conceptos básicos e.g. para la toma de decisiones y proceso de recompensas. Posteriormente estos conceptos se van a expandir a los Procesos Parcialmente Observables de Markov.

3.2.1. Procesos de Decisión de Markov

Los Procesos de Decisión de Markov (Markov Decision Processes, MDP por sus siglas del inglés), son un modelo donde un agente es el encargado de interactuar en sincronía con el mundo. El agente toma cómo entrada el estado del mundo y genera como salida

acciones que afectan el estado del mundo (ver Figura 3.2). Los MDP son la base para resolver problemas más complejos de decisiones. Un MDP es representado por la tupla $\langle S, A, T, R\rangle$ donde:

- S es un conjunto de estados $s \in S$.
- A es un conjunto de acciones con $a \in A$.
- T define la probabilidad $P(s_t|s_{t-1}, a_{t-1})$, es decir, la probabilidad de cambiar al estado s_t, dado que un tiempo anterior se estuvo en el estado s_{t-1} y se ejecutó la acción a_{t-1}. Formalmente, la función de transición entre estados que corresponde a la evolución de los estados a lo largo del tiempo se define como: $T : S \times A \to \Pi(S)$, y para las probabilidades de transición individuales $P(s_t|s_{t-1}, a_{t-1})$ [Cassandra, 1998].
- R define la recompensa esperada $r(s_t, a_t) \in R$, es decir, la recompensa que se obtiene al estar en el estado s_t y ejecutar la acción a_t. Ésta recompensa esperada puede ser positiva o negativa. Se define la función formalmente como $R : S \times A$. El problema es más complejo debido a la compensación entre las recompensas inmediatas a corto plazo con las recompensas que se producen en el futuro [Cassandra, 1998].

En general se quiere saber cuál es la mejor u óptima política para un MDP. El proceso de determinación de la política óptima generalmente se conoce como resolución del MDP. Esto implica que en cada punto en el tiempo hay que tomar una decisión, es decir, todo el problema es decidir qué acción tomar en un momento dado [Cassandra, 1998].

En este modelo, el siguiente estado s_t y recompensa esperada r_t dependen solo del estado previo s_{t-1} y la acción que fue elegida a_{t-1}. Esto es conocido como la *propiedad de Markov* [Kaelbling et al., 1998].

En los MDP se tiene algo que se llama *horizonte*, el cual se relaciona a que tantos pasos adelante se requiere computar. Éste horizonte puede ser finito o infinito. En la *optimización con horizonte-finito* el agente debe de maximizar la recompensa esperada los siguientes t-pasos. En la literatura, se considera que las recompensas del agente tienen un límite de tiempo de vida. Esto significa, que ahora se tiene un modelo conocido como *descuento de horizonte-infinito*. En éste modelo, se emplea un factor de descuento γ, sujetos

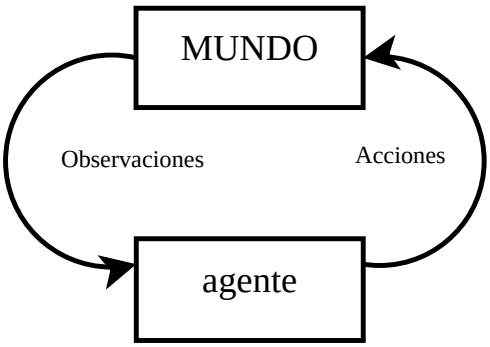


Figura 3.2: Interacción del agente con el mundo en los MDP

a $0 < \gamma < 1$. Éste factor de descuento asegura que la suma sea finita. La tarea del agente es optimizar las recompensas esperadas con el factor de descuento representadas por la expresión 3.1 donde r_t es la recompensa recibida en el paso t y γ es el factor de descuento, es importante notar que éste factor de descuento el valor más grande que puede tomar es aproximadamente a 1. Esta expresión es también la recompensa esperada para terminar con la ejecución de los procesos de decisión.

El horizonte-finito con muchos pasos por delante en la literatura es mejor conocido como planificación (planning) o predicción. Un problema de planificación (planning problem) es aquel en que dado un modelo completo y correcto de la dinámica del mundo y una estructura de recompensas, se debe de encontrar una manera óptima de su comportamiento. El término de horizonte-infinito es aplicado en el momento que un MDP completo converge a un autómata [Kaelbling et al., 1998, Cassandra, 1998].

$$E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{3.1}$$

Una política es una descripción del comportamiento de un agente. Esto se utiliza para determinar que acción se debe tomar en que momento de decisión. Existen dos tipos de políticas las cuales son *estacionarias* y *no-estacionarias*. En las políticas estacionarias, se especifica que en cada estado una acción debe de ser tomada independiente del tiempo t, es decir, $\pi : S \rightarrow A$. Ahora bien, una política no estacionaria, es una secuencia de situaciones dados por el tiempo t. La política π_t se utiliza para elegir la acción en el paso *t-ésimo*,

como una función del estado actual s_t. En un modelo con horizonte-finito las políticas no son estacionarias [Kaelbling et al., 1998, Cassandra, 1998]. De ahora en adelante nos enfocaremos en políticas *no-estacionarias* por nuestro tipo de problema a resolver.

Supongamos que dada una política, se busca evaluar el valor a largo plazo. Para esta tarea se utiliza $V_{\pi,t}(s)$ el cual corresponde a la suma esperada de las recompensas V obtenidas desde un estado inicial s ejecutando la política no-estacionaria π para t pasos. En el primer paso éste valor corresponde a la recompensa inmediata esperada que equivale a $R(s, \pi_1(s))$, para pasos más adelante se usa la ecuación 3.2 definida de manera recursiva.

$$V_{\pi,t}(s) = R(s, \pi_t(s)) + \gamma \sum_{s' \in S} T(s, \pi_t(s), s') V_{\pi,t-1}(s') \tag{3.2}$$

Al valor $V_{\pi,t}$ en la ecuación 3.2 se le conoce como *función valor*. Un enfoque de un método voraz dado una función valor, es tomar la acción que maximice la recompensa inmediata esperada [Kaelbling et al., 1998], ésta se muestra en la ecuación 3.3.

$$\pi_t^*(s) = \arg\max_a \left[R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_{t-1}^*(s') \right] \tag{3.3}$$

Toda esta información se encuentra disponible en Kaelbling, Littman y Cassandra et al. [Kaelbling et al., 1998] y Cassandra et al. [Cassandra, 1998].

Hasta este momento ya se tiene la formulación básica de los procesos de decisión, más adelante se van a presentar los MDP Parcialmente Observables, además de un conjunto de políticas de optimización.

3.2.2. Procesos de Decisión Parcialmente Observables de Markov

Los Procesos de Decisión Parcialmente Observables de Markov (Partially Observable MDP, POMDP por sus siglas del inglés) [Monahan, 1982] o también conocido como *MDP en el espacio continuo* [Kaelbling et al., 1998] son definidos por la tupla $\langle S, A, T, R, O, Z, \gamma, b_o \rangle$ de los cuales:

- S es un conjunto de estados $s \in S$.
- A es un conjunto de acciones $a \in A$.

- T define la probabilidad $P(s_t|s_{t-1}, a_{t-1})$, es decir, la probabilidad de cambiar al estado s_t, dado que un tiempo anterior se estuvo en el estado s_{t-1} y se recibio la acción a_{t-1}.
- R define la recompensa esperada $r(s_t, a_t) \in R$, es decir, la recompensa que se obtiene al estar en el estado s_t y se ejecutó la acción a_t (al igual que los MDP ésta recompensa puede ser positiva o negativa).
- O es un conjunto de observaciones con $o \in O$.
- Z define una probabilidad de observaciones $P(o_t|s_t, a_{t-1})$, es decir, la probabilidad de recibir una observación o_t, dado que estamos en el estado s_t y se recibio la acción a_{t-1}. Esta función se define formalmente como $O : A \times S \rightarrow \Pi(Z)$. Esta función mapea la acción en el momento t-1 y el estado en el tiempo t a una distribución sobre el conjunto de observación. Esto significa que la observación depende del estado resultante en la transición de estado. Se define $P(o_t|s_t, a_{t-1})$ para las probabilidades de observación individual [Cassandra, 1998].
- γ es un factor de descuento $0 \leq \gamma \leq 1$.
- b_0 es un estado de creencia inicial.

El funcionamiento del POMDP se puede resumir fácilmente de la siguiente manera, tomando en cuenta que en cada paso $t = 0, \ldots, T$, el mundo se encuentra en un estado s_t no observable. Entonces, debido a que s_t no es conocido completamente, existe una distribución entre todos los posibles estados. Esto se conoce como el *estado de creencia (belief state)* b_t. Para indicar la probabilidad de iniciar en un estado particular s_t en el estado de creencia, se utiliza la notación $b_t(s_t)$. En este proceso, se tiene algo que se llama *máquina de aprendizaje*. Ésta máquina es la encargada de seleccionar una acción a_t, recibir una recompensa r_t y una transición a un estado s_{t+1}, es entonces cuando la máquina recibe una observación o_{t+1}. Estos procesos se muestran en las Figuras 3.3 y 3.4. En estas figuras **SE** es la función del estimador de estado, el cual tiene como salida el nuevo estado de creencia b' [Kaelbling et al., 1998].

El estado de creencia podría ser descrito como el estado más probable del mundo, dada la experiencia pasada. Esto también equivale a decir que es la distribución de probabi-

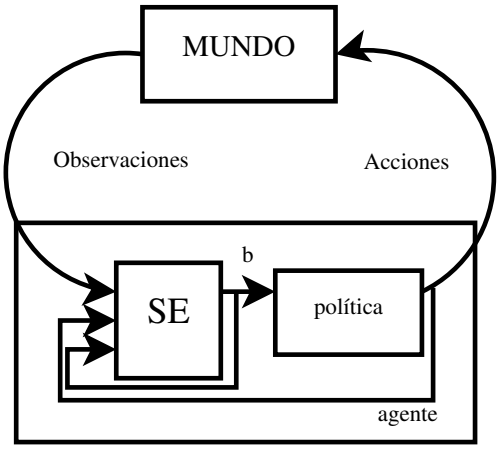


Figura 3.3: Interacción del agente con el mundo parcialmente observable

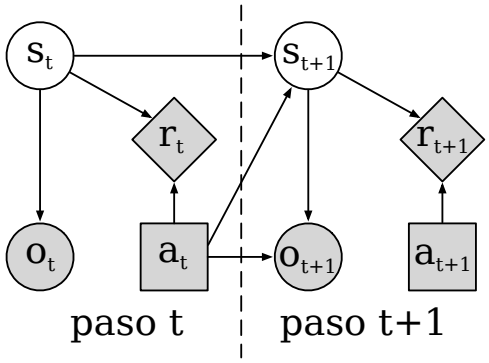


Figura 3.4: Estructura de un POMDP, imagen disponible en [Young et al., 2010]

lidades sobre los estados del mundo, donde la información que gana, es a través de la función de observación. El estado de creencia comprenden una estadística suficiente para la historia pasada y el estado inicial de creencia del agente [Kaelbling et al., 1998, Young et al., 2013, Cassandra, 1998, Monahan, 1982].

La manera de actualizar el estado de creencia b_{t+1}, está dado por un estado de creencia existente b_t, la última acción a_t, una nueva observación o_{t+1} de acuerdo a la siguiente ecuación.

$$b_{t+1}(s_{t+1}) = \eta P(o_{t+1}|s_{t+1}, a_t) \sum_{s_t} P(s_{t+1}|s_t, a_t) b_t(s_t) \tag{3.4}$$

En la ecuación 3.4, el valor de $\eta = \frac{1}{P(o_{t+1}|b_t,a_t)}$ es una constante de normalización. Las acciones están determinadas por la política π, la cual ha sido representada en la literatura por diferentes maneras. Esta política se representa de manera estocástica y mapea a una distribución de acciones $\pi(a|b) \in [0,1]$, donde $\pi(a|b)$ es la probabilidad de tomar la acción a en el estado de creencia b estando sujetos a $\sum_a \pi(a|b) = 1 \ \forall \ b$.

La suma de descuentos de las recompensas esperadas al iniciar en un estado b_t con la política π esta dado por la *función valor* $V^\pi(b_t) = E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$, la cuál en la literatura se expresa como la ecuación 3.5:

$$V^\pi(b_t) = \sum_{a_t} \pi(a_t|b_t) r(b_t, a_t) + \gamma \sum_{o_{t+1}} P(o_{t+1}|b_t, a_t) V^\pi(b_{t+1}) \tag{3.5}$$

La *función-Q* también conocida como $Q^\pi(b,a)$, proporciona la suma del valor de descuento esperado de las recompensas, para una acción a en un estado de creencia b, con la política π. Esto se puede simplificar como la ecuación 3.6.

$$V^\pi(b_t) = \sum_{a_t} \pi(a_t|b_t) Q^\pi(b_t, a_t). \tag{3.6}$$

La política optima π^* es aquella que maximiza V^π para obtener V^*, es decir:

$$V^*(b_t) = \max_{a_t} [r(b_t, a_t) + \gamma \sum_{o_{t+1}} P(o_{t+1}|b_t, a_t) V^*(b_{t+1})] \tag{3.7}$$

La ecuación 3.7 es conocida en la literatura como *la ecuación de optimización de Bellman*. Encontrar una política π que satisface ésta ecuación 3.7, es llamado resolver u optimizar el POMDP [Kaelbling et al., 1998, Cassandra, 1998].

Al resolver un POMDP se puede hablar de resolver una *función lineal convexa a pedazos* también conocida como función *híbrida*. Una función híbrida es una función que se define por múltiples sub-funciones, cada sub-función se aplica a un cierto intervalo del dominio de la función principal. Ésta función es lineal porqué es un característica de las recompensas del POMDP. Es a pedazos porqué de todas las políticas que se pueden generar no todas ofrecen la mayor recompensa sobre el estado de creencia, es decir, cada política es superior en un rango sobre el estado de creencia y esto genera la propiedad de ser convexa.

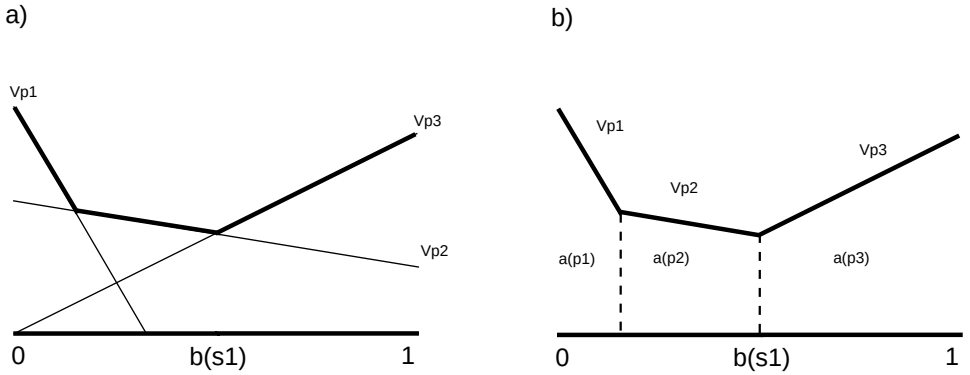


Figura 3.5: Políticas para la función híbrida, imagen disponible en [Kaelbling et al., 1998]

Por ejemplo, en la Figura 3.5.(a) se muestran tres políticas *Vp1*, *Vp2* y *Vp3* para el estado de creencia b del estado $s1$. En ésta Figura se muestra que la política *Vp1* solo se puede aplicar en una pequeña sección del estado de creencia, a diferencia de las otras dos políticas *Vp2* y *Vp3*. La línea negra a pedazos de esta figura 3.5.(a) es generada por las políticas y lo que representa es el rango de cada política utilizable sobre el estado de creencia. Ésta línea negra representa una función convexa, porqué si se seleccionan dos puntos de la función y se unen con una línea recta, todos los puntos de la función son menores o igual a la línea recta. Ésta simple prueba es la característica de convexidad. Entonces, el trabajo ahora es detectar ese rango de cada política utilizable para llegar a la Figura 3.5.(b). Con todo lo que se ha mencionado de las secciones anteriores, se puede decir que solo nos interesa encontrar la mejor política, en el caso de la Figura 3.5.(b) es la política *Vp3* porqué, es la que tiene un mayor rango del estado de creencia. En éste ejemplo solo se tienen pocas políticas de las cuales todas son utilizadas, pero no siempre es así porque hay ejemplos con una gran cantidad de políticas donde la gran mayoría no son requeridas. En estos casos se aplica alguna técnica para *poda de políticas*, la cual permite encontrar un conjunto mínimo utilizable que puede representar a la función híbrida, en ese momento se conoce cómo *función parsimoniosa* [Kaelbling et al., 1998, Cassandra, 1998].

La Figura 3.6 muestra el proceso de planificación desde un estado inicial S_0 hasta un estado final S_t, donde la t significa t-pasos adelante. En ésta figura se muestra el funcionamiento de los POMDP en la capa 'Oculto', 'Visible' y 'Estado de creencia'. Es decir, la

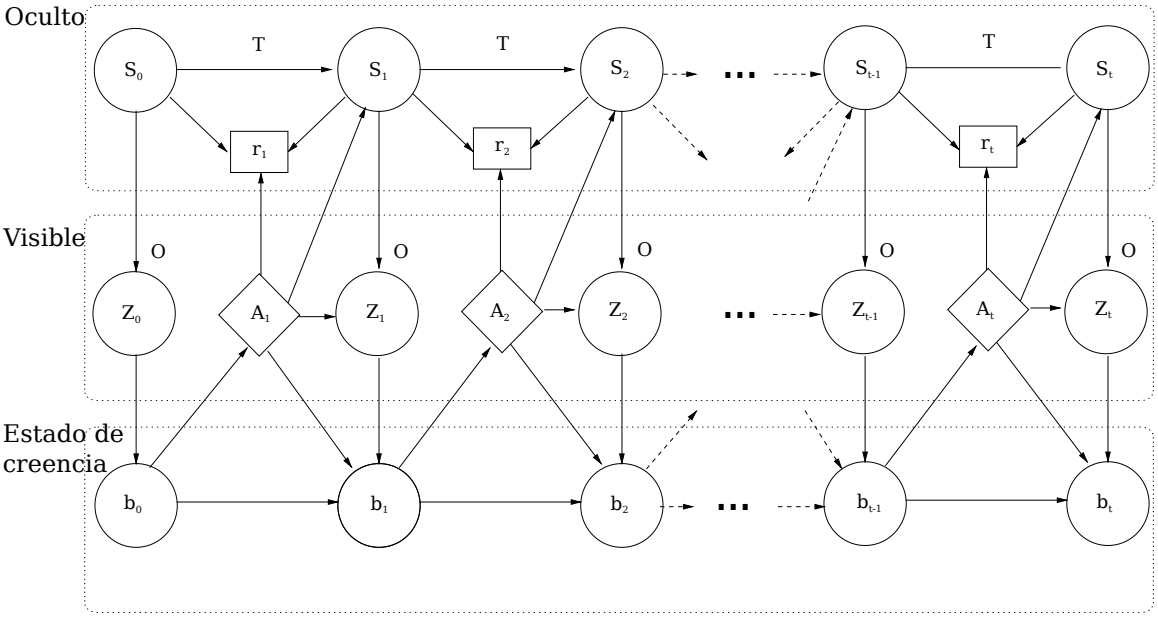


Figura 3.6: Ilustración de la planificación con POMDP

probabilidad de que el agente se pueda equivocar, lo que observa el agente en ese instante de tiempo y el estado completo del mundo respectivamente.

En la capa 'Oculto' se muestra que la actualización del estado S_t solo depende del estado anterior S_{t-1} y la acción A_t, y debido a esta transición entre estados se obtiene una recompensa r_t y una observación O. En la capa 'Visible' se muestra cómo la observación Z_t depende del estado S_t y la acción A_t. Finalmente, la tercer capa muestra la actualización del estado de creencia b_t, el cual depende del anterior estado de creencia b_{t-1}, la acción A_t y la observación Z_t. Ésta figura es un buen ejemplo en el cuál se puede apreciar completamente el funcionamiento del POMDP durante el proceso de planificación.

Para finalizar ésta sección se va a mostrar un ejemplo, en Kaelbling, Littman y Cassandra [Kaelbling et al., 1998] se encuentra el problema más utilizado y sirve para mostrar el funcionamiento de los POMDP de una mejor manera. Éste problema es conocido como el *problema del tigre* el cual consiste en que hay dos puertas y en alguna de ellas se encuentra encerrado un tigre. Las acciones que se pueden realizar son *escuchar* para ganar información si detrás de esa puerta se encuentra o no el tigre, o bien *abrir la puerta izquierda* o *abrir la puerta derecha*. Al realizar cada una de estas acciones se obtiene un conjunto de

Tabla 3.1: Problema del tigre

-	Acción	Estado	Tigre Izquierda (TI)	Tigre Derecha (TD)
Transiciones	Abrir Izq.	TI	0.5	0.5
		TD	0.5	0.5
	Escuchar	TI	1.	0.
		TD	0.	1.
	Abrir Der.	TI	0.5	0.5
		TD	0.5	0.5
Observaciones	Abrir Izq.	TI	0.5	0.5
		TD	0.5	0.5
	Escuchar	TI	0.85	0.15
		TD	0.15	0.85
	Abrir Der.	TI	0.5	0.5
		TD	0.5	0.5
	Acción		Tigre Izquierda (TI)	Tigre Derecha (TD)
Recompensas	Abrir Izq.		-100	10
	Escuchar		-1	-1
	Abrir Der.		10	-100

recompensas que pueden ser positivas o negativas. En la Tabla 3.1 se muestran los valores de las transiciones entre estados, observaciones y recompensas asociadas a éste problema.

La Figura 3.7 muestra la recompensa inmediata (1 paso adelante), cómo se puede observar la función híbrida (línea negra) depende directamente de las recompensas. Ésta función es fácil de gráficar porqué en éste ejemplo se tienen dos estados y tres acciones, con esto se utilizan dos dimensiones, si fueran tres estados entonces se tienen tres dimensiones y así sucesivamente. En la figura 3.7 la línea azul, verde y roja representan las políticas de *abrir-derecha*, *abrir-izquierda* y *escuchar* respectivamente.

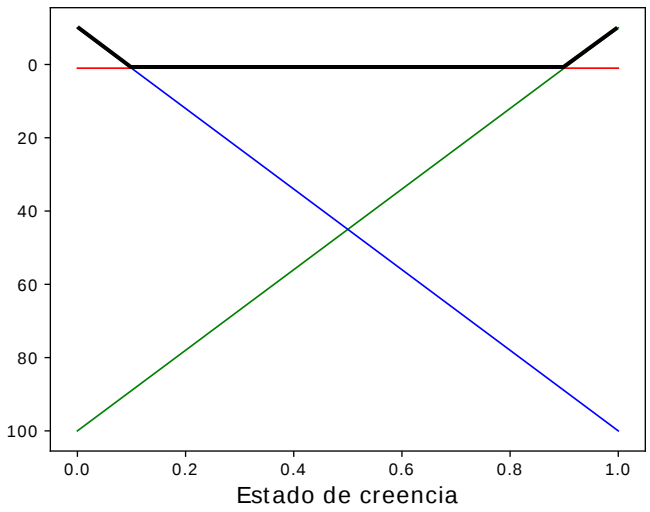


Figura 3.7: Recompensa inmediata (un paso adelante)

3.3. Trabajo Relacionado

En la actualidad han surgido varias áreas que se encargan de modelar las emociones, entre ellas se encuentra el *Cómputo afectivo*. Los investigadores de ésta área, están interesados en cuatro problemas principales, los cuales abarcan i.e. (1) Reconocimiento de las emociones. (2) Generación de señales moduladas con afecto, es decir, señales de voz con emociones, o simulación de expresiones faciales con imágenes. (3) Estudio de las emociones humanas incluyendo la interacción afectiva y adaptación y (4) Modelado de la interacción afectiva entre Humano-Computadora [Hoey et al., 2016]. Este trabajo de tesis se enfoca en los puntos 1, 3 y 4, debido a que se reconoce el estado emocional del parlante por medio de la voz. Además, los modelos emocionales son generados a base de conversaciones afectivas y finalmente, este sistema aplicado en una conversación real, una computadora puede estar estimando que emociones se encuentran presentes qué en momento. De ésta manera se puede monitorear el estado emocional del parlante, es decir, se puede observar como su estado emocional se comporta por estimulos de otros parlantes, o bien del medio.

Los autores Hoey, Schröder y Alhothali et al. [Hoey et al., 2016] basan el reconocimiento afectivo a un sistema inteligente interactivo, usando teorías establecidas del

razonamiento emocional. Con estas ideas, surge el método *Procesos de Control Afectivo* (Affect Control Processes, ACP por sus siglas del inglés), el cual ha sido exhaustivamente trabajado en los últimos años, sin embargo, aún no es tan conocido en el área del cómputo afectivo. ACP trabaja con un vector de tres valores, el cual es conocido como EPA. Éste vector contiene las características lingüísticas para determinar el espacio afectivo. Los valores de éste vector corresponden i.e. E=Meta de un evento, P=El potencial del agente, A=La urgencia implicada por alguna situación. ACP trata la dinámica de los estados emocionales y comportamientos como trayectorias continuas en un espacio afectivo.

Los autores Hoey, Jesse y Schröder et al. [Hoey and Schröder, 2015] han trabajado en otra aproximación que utiliza los conceptos de ACP. Ésta es conocida como *Self-ACP*. La diferencia entre estas dos, es que Self-ACP en algún momento el usuario puede cambiar de perspectivas durante la interacción.

La similitud de este trabajo de tesis con ACP, es que ambos se basan en estimar el estado emocional con características de los parlantes que dependen de su entorno. La diferencia de este trabajo de tesis con ACP, es que se realiza la tarea de predicción utilizando sólo el estado emocional que se obtiene a través de las señales de voz. En este trabajo de tesis no se evalúa a los usuarios ni el contexto, solamente se enfoca en la respuesta dado el tono de la frase que fue percibido. Ésta actividad se realiza de ésta manera, debido a que se ha observado que en una conversación real, es muy común responder dependiendo del tono en que se nos esta hablando y a su vez importa el contexto relacionado en las frases.

3.4. Determinación del estado actual emocional

En esta sección se menciona el trabajo desarrollado para la estimación del estado emocional enfocado al conjunto de estados, observaciones y recompensas para poder utilizar los POMDP. En el sistema implementado lo que se obtiene son observaciones emocionales del medio. Es decir, las emociones que se perciben de algún parlante, y lo que sale es una estimación del completo estado emocional de ese parlante. La Figura 3.8 muestra un ejemplo de la interacción de este sistema durante una conversación. En ésta imagen se puede observar que el trabajo realizado en la sección anterior es importante para conocer con una mayor

exactitud el estado emocional que se percibe de un parlante en ese instante de tiempo.

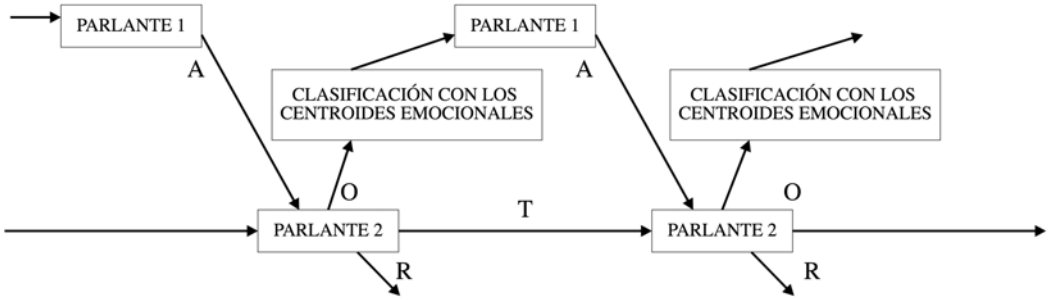


Figura 3.8: Diagrama del sistema implementado para la estimación del estado emocional

La primera aplicación que se desea desarrollar con este trabajo de tesis es un predictor de emociones que de sólo seguimiento al estado de felicidad para evaluar un juego a través de la voz. En éste juego se va a detectar cuales son las mejores configuraciones de las partidas con las que el jugador tiene una mejor empatía. En caso de que se detecte al jugador aburrido o enojado el sistema debe de hacer lo mejor posible para cambiar el estado emocional del jugador utilizando diferentes escenarios, música de fondo, luces y todo lo que sea posible para captar la atención del jugador. Ahora bien, en caso de una conversación las acciones es hacer expresiones emocionales agradables para hacer que el parlante cambie a un estado emocional de felicidad.

Entonces, una vez mencionado lo que se desea desarrollar, en las secciones anteriores se ha visto que en los procesos de decisión se requieren de algunos componentes principales que en su implementación se pueden ver como matrices. Estos componentes son las transiciones, observaciones y recompensas. Para obtener estas matrices se analizó una cantidad de audios que contienen conversaciones emocionales disponibles en el internet incluyendo el audio de las aventuras del mago de Oz entre otros audios de las bases de datos de las cuales se dispone (EMO-DB y SAVEE-DB). Para crear éstas matrices se implementa la idea de las cadenas de Markov donde se cuentan las transiciones de un estado emocional anterior s_{t-1} al estado emocional actual s_t. En este trabajo de tesis no se tiene un estado emocional especifico de inicio, entonces el estado de creencia inicial b_0 tiene una distribución uniforme sobre todos los estados emocionales. Finalmente, el último componente es un

factor de descuento el cual corresponde al valor de $\gamma = 0.95$. Éste valor se eligió porqué es más utilizado en la literatura.

La matriz de transición fue creada suponiendo que si algún parlante se encuentra en un estado emocional aleatorio, por ejemplo enojo y si recibe un estimulo con ese mismo estado emocional, entonces la transición al siguiente estado emocional de ese parlante aun es el enojo. Sin embargo, si en lugar de recibir un estimulo relacionado a su estado actual, éste parlante recibe un estimulo diferente sin importar cual sea, entonces se tiene una distribución uniforme sobre todos los estados emocionales del parlante (éste planteamiento se relaciona al problema del tigre). El porqué de ésta suposición es que todas las personas reaccionan de diferentes maneras y aun no se tiene un modelo uniforme de las reacciones de las personas dadá alguna situación.

La matriz de observación se construye a partir la Tabla 3.2 que se obtuvo de varias conversaciones emocionales concentrando las emociones que se eligieron como base. Y las que se encuentran disponibles de las bases de datos seleccionadas, las cuales son: Enojo, Aburrimiento, Disgusto, Ansiedad/miedo, Felicidad, Tristeza, Sorpresa y Neutral. Ésta tabla muestra las observaciones de los parlantes, si algún parlante se encuentra en algún estado inicial s_t de la primera columna y si él parlante recibe como estimulo alguna emoción de la misma primer columna. El siguiente estado s_{t+1} emocional que se encuentre, corresponde a la segunda columna de cada tabla. Es decir, se tiene una conversación donde existe un parlante que se encuentra enojado, y él en diferentes casos puede recibir como estimulo alguna de las ocho emociones anteriores. El conjunto de estados emocionales al que éste parlante puede cambiar es a enojo y neutral con la misma probabilidad, es decir, solo puede seguir enojado o se puede tranquilizar.

En nuestro ejemplo se mencionó que este trabajo de tesis se interesa en que algún parlante llegue al estado emocional de felicidad, por este motivo las recompensas se van a diseñar de tal manera en que un estado neutro o positivo se gane más a diferencia de un estado negativo. Entonces, la matriz de recompensas se diseñó en que un agente independiente del estado en que se encuentre gana una recompensa negativa (-1) cada vez que llega a un estado de enojo, aburrimiento, disgusto, ansiedad, miedo, tristeza, es decir, algún estado con una valencia negativa. Ahora bien, éste agente gana una recompensa positiva cuando

Tabla 3.2: Tabla de observaciones

Estado s_t	Estado s_{t+1}
Enojo	Enojo, Neutral
Aburrimiento	Aburrimiento, Neutral
Disgusto	Enojo, Disgusto, Neutral
Ansiedad-Miedo	Enojo, Ansiedad-Miedo, Tristeza, Neutral
Felicidad	Felicidad, Neutral
Tristeza	Tristeza, Neutral
Sorpresa	Miedo, Feliz, Neutral, Sorpresa
Neutral	Enojo, Aburrimiento, Disgusto, Ansiedad-Miedo Felicidad, Tristeza, Sorpresa, Neutral

se cambia a un estado neutro (1) y especialmente gana más cuando cambia a un estado de valencia positiva (2). El porqué de estos valores es que se quiere un aumento gradual en las transiciones emocionales. Ahora bien, si en lugar de querer emociones positivas se quiere un juego, donde la idea es que el jugador experimente miedo. Entonces la matriz de recompensas deberá ser modificada de tal manera que cada vez que se experimente la emoción de miedo se gane más.

La actualización del estado de creencia en los POMDP (Ecuación 3.4) permiten desarrollar el seguimiento del estado emocional. Ésta actividad se logra al monitorear a través del tiempo el estado de creencia. En este trabajo de tesis, el estado de creencia representa el conjunto de estados emocionales utilizados. La idea del seguimiento es mostrar en cada instante los niveles emocionales de los parlantes y cómo estos van cambiando durante alguna conversación a través del diálogo. Un ejemplo se muestra en el siguiente capítulo.

3.5. Comentarios finales

En este capítulo se mencionó en que consisten los POMDP desde crear políticas, obtener la función lineal convexa y actualización del estado de creencia. En éste trabajo de tesis la actualización del estado de creencia se utiliza para estar monitoreando la distribución de las emociones en un sistema de tiempo real. Ahora bien, si se desea predecir cual es la emoción que será dominante un tiempo adelante esto se menciona en el siguiente Capítulo

4. En éste Capítulo 4 también se mencionan algunas técnicas de optimización y algoritmos para la poda de políticas. Finalmente, se menciona los Procesos de Decisión de Markov con la optimización de Monte Carlo.

Capítulo 4

Predicción del estado emocional

En el capítulo anterior se presentaron los Procesos de Decisión Parcialmente Observables de Markov además de un ejemplo de su funcionamiento. En el capítulo anterior también se mencionó que para monitorear el estado emocional se utiliza el estado de creencia. Realizar el monitoreo de las emociones de ésta manera permite observar cómo cambian a través de una conversación los niveles emocionales de los parlantes. Ahora en este capítulo se van a mencionar algunas técnicas de optimización utilizadas para realizar la predicción con los POMDP.

4.1. Introducción

En esta sección se va a mencionar porqué es necesario utilizar técnicas de optimización con los POMDP. En los POMDP la manera para crear, ejecutar y eliminar políticas, así como la cantidad de estados que tenga el problema a resolver, que tantos pasos se quieren predecir y la manera de actualizarse son parámetros que afectan directamente en el redimiendo. Por ésta razón, varios investigadores han estado desarrollando estrategias para realizar ésta actividad de la predicción lo más exacto y rápido posible. Entre esas estrategias se encuentra cómo obtener la función parsimoniosa rápido y actualizar el estado de creencia cuando se tiene una gran cantidad de estados del problema a resolver.

Los algoritmos más utilizados para realizar la predicción son: Enumeración Exhaustiva, Poda Incremental, el algoritmo Dos Pasadas, el algoritmo del Testigo. Además

del algoritmo con la optimización de Monte-Carlo y POMDP (POMCP). Con estos algoritmos se busca mostrar el desempeño en el sistema desarrollado para este trabajo de tesis aplicado en tiempo real. Todos estos algoritmos se implementaron en este trabajo de tesis con el lenguaje de programación Python y el software de programación lineal Gurobi con la API para Python. Gurobi es un software para resolver problemas LP. Utilizar un software especializado para problemas LP es porqué varios autores mencionan que la mayoría del tiempo un POMDP lo dedica al problema LP que se relaciona a la función parsimoniosa. El objetivo de este capítulo es mostrar de una manera sencilla los algoritmos utilizados para la predicción con los POMDP.

4.2. Políticas de Optimización

La optimización más común utilizada durante la etapa de construcción de políticas se basa en calcular la función valor (ver apéndice A, ecuación A.1), una por acción a la vez y luego fusionar los conjuntos de políticas que se obtienen. Ésta tarea permite utilizar técnicas que no serían directamente aplicables si se intentara construir el conjunto de políticas utilizables directamente. El algoritmo del testigo fue el primero en utilizar éste enfoque de la función valor [Cassandra, 1998].

A continuación se va a describir la idea central de dos algoritmos que son interesantes en su funcionamiento, a pesar de que no se implementaron en este trabajo de tesis, se mencionan porque algunos algoritmos implementados utilizan algunas características básicas de ellos: En primer lugar se tiene el algoritmo de una pasada (The one pass algorithm). Éste algoritmo de una pasada funciona identificando regiones una por una de la función valor siendo capaz de crear un conjunto de restricciones que forman el borde de la región óptima. Después, se buscan las fronteras y se determina si existe alguna otra región más allá del borde. Aunque ésta acción evita numerar exhaustivamente todo el conjunto de políticas útiles por iteración, éste algoritmo de una pasada es lento [Young et al., 2013, Cassandra, 1998]. En segundo lugar se tiene el algoritmo de soporte lineal de Cheng (The support lineal Cheng algorithm). Éste algoritmo de soporte lineal basa su funcionamiento en buscar puntos sobre las esquinas de las regiones de la función valor, es decir, se selecciona un punto arbitrario

del estado de creencia y se genera la políticas para ese punto, luego se verifica la región de esa política si es la correcta en todas las esquinas del estado de creencia. En caso de no ser la misma región, entonces la mayor diferencia entre las dos políticas debe estar en uno de los puntos del borde de la región de creencias donde se denomina esa política. La clave de éste algoritmo se encuentra en que si la función valor es incorrecta, la mayor diferencia se producirá en una esquina; y si se genera en todas las esquinas posibles de la región, se tiene la certeza de que éste algoritmo no fallara. Éste algoritmo de soporte lineal de Cheng es eficiente debido a que tiene características de interpretación geométrica, las cuales son una buena estrategia en problemas con alta dimensión [Young et al., 2013, Cassandra, 1998].

Ahora se va a comentar la idea fundamental de los algoritmos implementados en este trabajo de tesis, para mayor información se puede consultar el apéndice A.

Valor iteración

El algoritmo valor iteración es un algoritmo fácil de aplicar, que permite estimar una política mientras se tenga una suficiente exploración de todas las rutas que se pueden tomar. Los estados de creencia del algoritmo de Valor iteración son vistos como vectores de características arbitrarios. La dinámica se estima directamente en el espacio de características más representativo [Young et al., 2013]. Además de que éste algoritmo sirve como base para encontrar políticas en el caso de MDP parcialmente observables [Kaelbling et al., 1998].

Enumeración exhaustiva

El algoritmo enumeración exhaustiva [Monahan, 1982] construye una larga representación de las políticas para después realizar la poda. La ventaja de éste algoritmo es que solamente se almacenan las políticas útiles por las acciones del nivel del paso-(t-1). Éste algoritmo es lento porque el tiempo de computo que se invierte se divide en dos etapas, las cuales son generación y poda de los árboles de políticas. La etapa de poda de políticas requiere resolver un problema de programación lineal [Kaelbling et al., 1998].

El algoritmo testigo (The Witness Algorithm)

El funcionamiento del algoritmo del testigo es similar al algoritmo de enumeración exhaustiva y el algoritmo de soporte lineal de Cheng, esto se debe a que el algoritmo del testigo define regiones para una política y busca un punto donde esa política no sea dominante. Éste algoritmo del testigo no se preocupa por todas las acciones todo el tiempo y se concentra en encontrar la mejor función valor para cada una de las acciones por separado, además que solo se preocupa por una observación a la vez. En el momento en que el algoritmo del testigo encuentra las funciones valor, todas estas funciones son combinadas en una función valor final.

Algoritmo de Poda incremental (Incremental Pruning Algorithm)

El funcionamiento del algoritmo de poda incremental combina elementos del algoritmo del testigo y el algoritmo de enumeración exhaustiva pero con una mayor velocidad. El incremento de la velocidad de éste algoritmo se obtiene debido a que se eliminan repetidamente los árboles de políticas no útiles durante el procedimiento de generación. Por parte del algoritmo del testigo se considera la construcción del conjuntos de políticas para cada acción individualmente y luego se centra en cada observación a la vez. La idea básica es eliminar todas las regiones adyacentes, dado que el problema principal es encontrar todas las diferentes combinaciones de estrategias futuras. Entonces, para una acción dada, primero se construyen todos los conjuntos de estados, acciones y observaciones (uno para cada observación). Luego se eliminan todas las políticas que no aportan información y las políticas que restan son combinados nuevamente. Esto se repite para otra acción y luego los conjuntos combinados de manera similar al algoritmo del testigo.

Algoritmo de dos pasadas (The two pass algorithm)

El funcionamiento del algoritmo de dos pasadas es similar al algoritmo de una pasada y el algoritmo del testigo, de hecho éste algoritmo de dos pasadas tiene éste nombre porqué, al igual que el algoritmo de una pasada al inicio recorre el espacio de información para construir la función valor una a la vez por acción. La segunda pasada se utiliza para

fusionar estos conjuntos de políticas que se obtienen [Cassandra, 1998].

Optimización Monte-Carlo

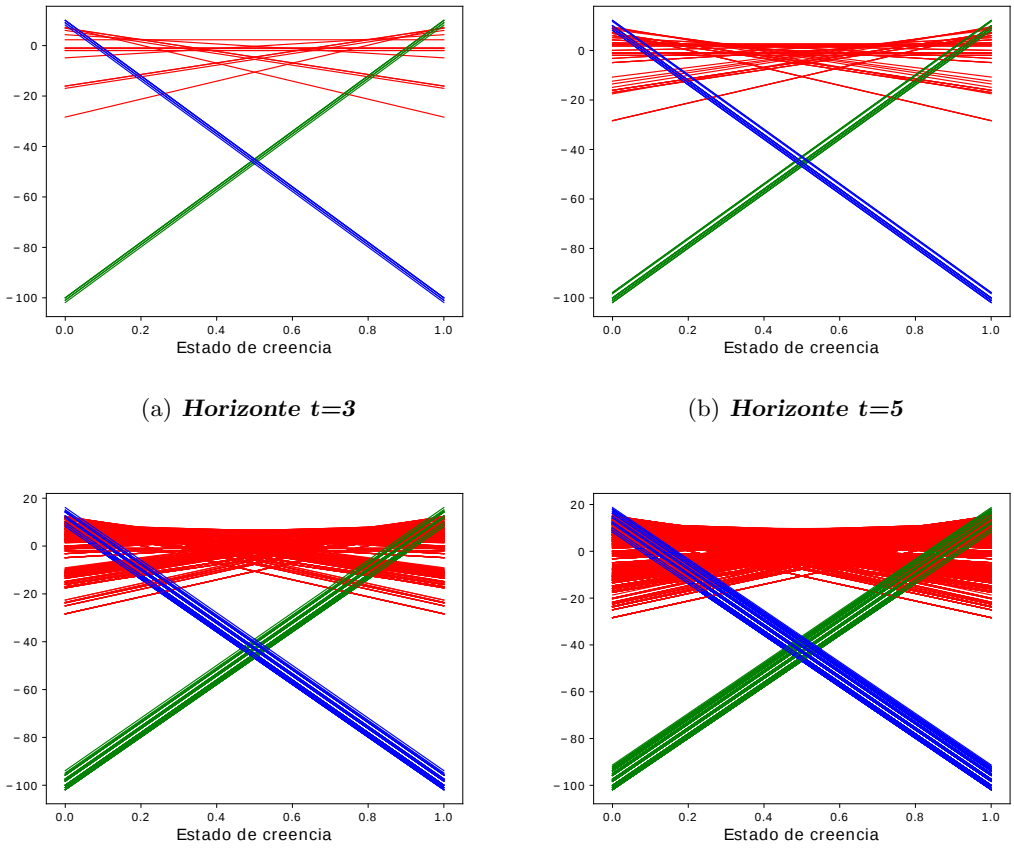
Hasta este momento la mayoría de métodos que se han mencionado son *Fuera de línea* y *exactos*. Además de que la mayoría de los métodos de planificación anteriormente descritos se basan en actualizar el estado de creencia por medio del Teorema de Bayes. Ésta estrategia con problemas que tienen una gran cantidad de estados, acciones y observaciones, un sola actualización puede resultar costoso en tiempo de computo, por la cantidad de operaciones a realizar [Silver and Veness, 2010]. Existen problemas con una gran cantidad de estados que son conocidos como POMDP largos y para llegar a su solución se requieren de algoritmos eficientes para disminuir el tiempo de computo. Para realizar ésta actividad se integra el POMDP con la optimización de Monte-Carlo la cual se conoce cómo POMCP. Éste método combina la actualización del estado de creencia del agente con el árbol de búsqueda Monte-Carlo (Monte-Carlo Tree Search, MCTS por sus siglas del inglés) del actual estado de creencia.

4.3. Algunos Ejemplos

En esta sección se van a mencionar algunos ejemplos comenzando con el problema del tigre descrito en el capítulo anterior (Capítulo 3). En el capítulo 3 se mostró cómo se obtiene la recompensa inmediata, ahora si se mira un horizonte más adelante, por ejemplo a un horizonte de unos 3, 5, 10 y 15 pasos adelante cómo se muestra en las Figuras 4.1.(a), 4.1.(b), 4.1.(c) y 4.1.(d) respectivamente. En estas figuras se puede observar que encontrar la función híbrida en cada paso comienza a complicarse. Cada vez que ocurre este problema es donde se utilizan las técnicas de optimización para poder encontrar la política optima.

En la literatura de los POMDP existe una amplia cantidad de problemas que han sido utilizados para mostrar su funcionamiento. Estos problemas contienen diferentes cantidades de estados, acciones y observaciones. Todos estos se encuentran disponibles en la pagina www.pomdp.org con la etiqueta de 'problemas de juguete'.

Los ejemplos que se utilizaron para medir la eficiencia tienen diferente cantidad de



(a) ***Horizonte t=3***

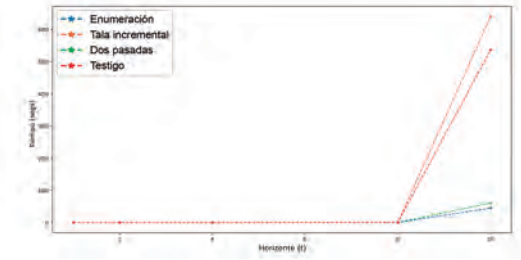
(b) ***Horizonte t=5***

(c) ***Horizonte t=10***

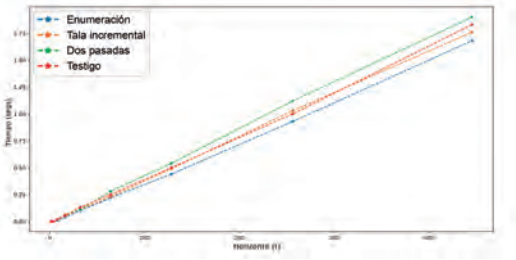
(d) ***Horizonte t=15***

Figura 4.1: Problema del tigre

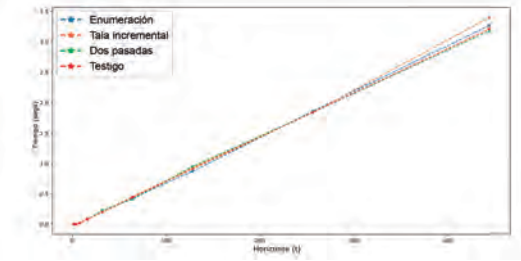
estados, acciones y observaciones. Con estas variables se puede determinar su comportamiento cómo se muestra en la Figura 4.2. En ésta figura se muestra una planificación a largo plazo hasta encontrar la convergencia del problema como se realiza en [Kaelbling et al., 1998]. Lo que se puede observar de la Figura 4.2 es que el algoritmo con el mejor rendimiento se lo lleva la enumeración exhaustiva, en segundo lugar le continua tala incremental, en tercer lugar el algoritmo del testigo y en el último lugar se encuentra el algoritmo de las dos pasadas. En especial éste comportamiento se relaciona al tiempo que interviene en la creación de las políticas y para encontrar el conjunto mínimo de las políticas que representa la función híbrida.



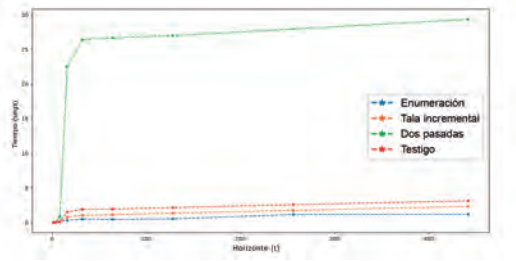
(a) ***Prob. 4x3.95*** $|S|$***=11,*** $|A|$***=4,*** $|O|$***=6***



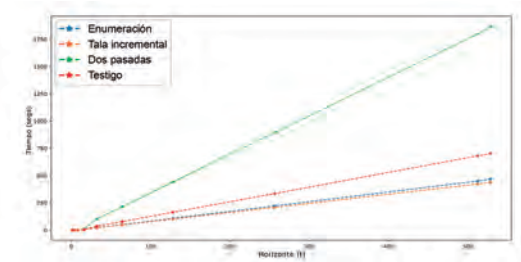
(b) ***Prob. 4x4.95*** $|S|$***=16,*** $|A|$***=4,*** $|O|$***=2***



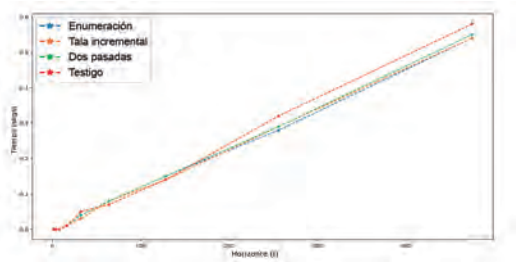
(c) ***Prob. cheese.95*** $|S|$***=11,*** $|A|$***=4,*** $|O|$***=7***



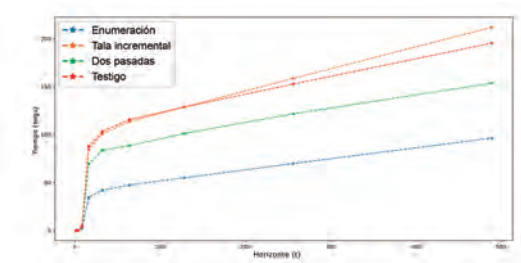
(d) ***Prob. hanks.95*** $|S|$***=4,*** $|A|$***=4,*** $|O|$***=2***



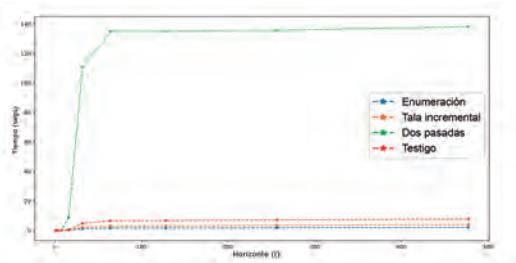
(e) ***Prob. network.95*** $|S|$***=7,*** $|A|$***=4,*** $|O|$***=2***



(f) ***Prob. parr95.95*** $|S|$***=7,*** $|A|$***=3,*** $|O|$***=6***



(g) ***Prob. shuttle.95*** $|S|$***=8,*** $|A|$***=3,*** $|O|$***=5***



(h) ***Prob. tiger.95*** $|S|$***=2,*** $|A|$***=3,*** $|O|$***=2***

Figura 4.2: Ejemplos de Juguete

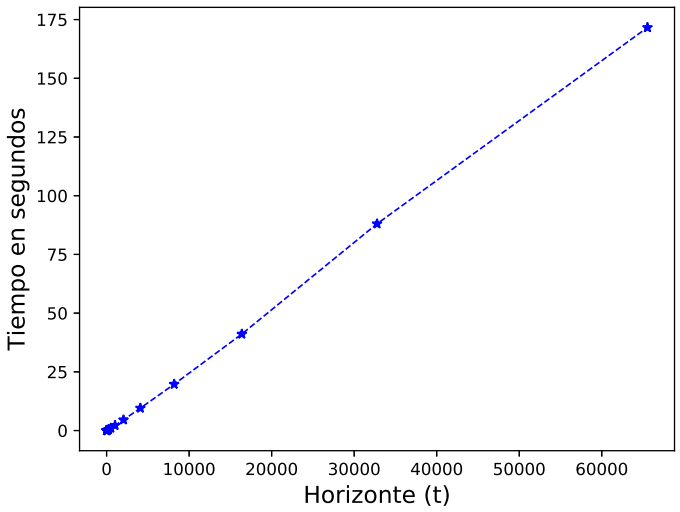


Figura 4.3: Desempeño del algoritmo POMCP con el problema Pocman

La Figura 4.3 muestra un ejemplo del desempeño con el algoritmo POMCP que se obtiene del problema 'Pocman' [Kyriakos,]. Pocman es una versión parcialmente observable del popular juego de arcade Pacman. El agente pacman se mueve en un laberinto recolectando comida mientras es perseguido por fantasmas. El juego termina cuando Pacman recoge toda la comida o es atrapado por un fantasma. En la versión parcialmente observable, en cierto sentido se juega al pacman, en éste juego no se puede observar todo el laberinto, las

Tabla 4.1: Ejemplos de juguete

	Problema	N. Estados	N. Acciones	N. Observaciones
a)	4x3.95	11	4	6
b)	4x4.95	16	4	2
c)	cheese.95	11	4	7
d)	hanks.95	4	4	2
e)	network.95	7	4	2
f)	parr95.95	7	3	6
g)	shuttle.95	8	3	5
h)	tiger.95	2	3	2

posiciones de fantasmas ni los alimentos. El Pocman recibe 10 bits en observación correspondientes a sus sentidos: cuatro bits en observación para su vista, que indican si puede ver un fantasma en cada una de las cuatro direcciones cardinales; un bit en observación que indica si puede escuchar un fantasma, 4 bits en observación que indican si puede sentir una pared en esta dirección; finalmente, una observación para oler alimentos, indicando la existencia de comida en las posiciones de cuadrícula adyacentes (diagonal y cardinal). El número de estados es aproximadamente 10^{56}, hay 4 acciones y 1024 observaciones. El horizonte de éste problema a 65,536 pasos adelante, el tiempo promedio que se obtiene es 2.85 minutos con 1000 simulaciones de la misma manera en que lo realizo el autor Thrun [Thrun, 2000]. Suponiendo un horizonte de 15 pasos adelante cómo en las pruebas anteriores, se obtiene un tiempo 0.0223874 segundos el cual es muy poco para la cantidad de estados que se están utilizando.

4.4. Implementación del predictor del estado emocional

En ésta sección se mencionan las pruebas realizadas para la predicción del estado emocional con los POMDP. En el Capítulo 3 se mencionó acerca del conjunto de estados, acciones y recompensas que se utilizaron en éste problema. Además que el monitoreo de las emociones se obtiene al actualizar el estado de creencia a través del tiempo.

El trabajo que se desarrolló en esta sección. Primero es conocer el funcionamiento de los POMDP con algunas técnicas de optimización y programar los cinco algoritmos antes mencionados. Además de realizar pruebas con los modelos emocionales que se mencionaron anteriormente. Segundo es observar el comportamiento de los algoritmos exactos y aproximados en un sistema de tiempo real resolviendo problemas con un horizonte a corto-mediano plazo. La programación se realizó con el lenguaje de programación Python y Gurobi.

En este trabajo de tesis se considera que el tiempo promedio que toma el intercambio de frases es de aproximadamente un segundo dentro de una conversación. La predicción se considero realizarla con 3, 5, 10 y 15 pasos adelante, porqué la idea es utilizar este sistema en aplicaciones como en juegos o bien centros de llamadas inteligente donde no se requiere una predicción a largo plazo. Además, cómo las bases de datos utilizadas comparten emocio-

nes similares se decidió juntarlas para la predicción, es decir las emociones que se utilizaron i.e. Enojo, Disgusto, Miedo, Felicidad, Neutral, Tristeza y Sorpresa.

La Tabla 4.2 corresponde a las pruebas realizadas para la planificación utilizando los algoritmos exactos con los POMDP, los valores se expresan de la forma ⟨Tiempo en segundos,Nº de políticas⟩ en ese horizonte. En ésta tabla se muestra que el algoritmo del testigo obtiene el mejor desempeño a diferencia del algoritmo de las dos pasadas con el peor desempeño durante la predicción. En estas tablas también se muestra que el algoritmo de poda incremental compite con el algoritmo de testigo, de hecho lo dos tienen un desempeño similar en la sección a). La sección b) de estas tablas muestra una predicción a largo plazo, que por las características del problema se acerca a la convergencia, es decir, donde las nuevas políticas que son generadas en el paso t son muy similares a las políticas del paso anterior *t-1*.

Al realizar éstas pruebas se observó que varios de éstos algoritmos empiezan con una buena velocidad para realizar la tarea donde se crean y podan políticas. Sin embargo, al acercarse a la convergencia se comienza a decrementar su velocidad. En éstos últimos pasos se muestra que el algoritmo que proporciona el mejor rendimiento es enumeración exhaustiva, es decir, mejoró al transcurso del tiempo durante la predicción siempre y cuando se encuentra cerca de la convergencia. Éste efecto es interesante porqué el algoritmo de enumeración exhaustiva no implementa alguna técnica de optimización especifica, debido a que su funcionamiento se centra en generar todo y podar manteniendo la mejor política del paso anterior. Por otra parte, el algoritmo de dos pasadas desde el inicio requiere de más tiempo en comparación con enumeración exhaustiva, tala incremental y testigo. Además, se puede verificar que a lo largo de la planificación nunca mejora en velocidad. La sección c) de estas tablas muestra el tiempo promedio y la cantidad de políticas que se obtienen en el momento qué los algoritmos implementados llegan a la convergencia con un error 2.56e-11.

La Tabla 4.3 muestra un rango de tiempo al utilizar los POMCP para resolver el problema de la predicción de las emociones. En ésta tabla se muestra que la velocidad con éste método es menor en comparación con los cuatro algoritmos anteriores i.e. numeración exhaustiva, tala incremental, dos pasadas y testigo utilizando una sola simulación. Es fácil notar que aumentar el número de simulaciones va a aumentar el tiempo pero se va a ganar

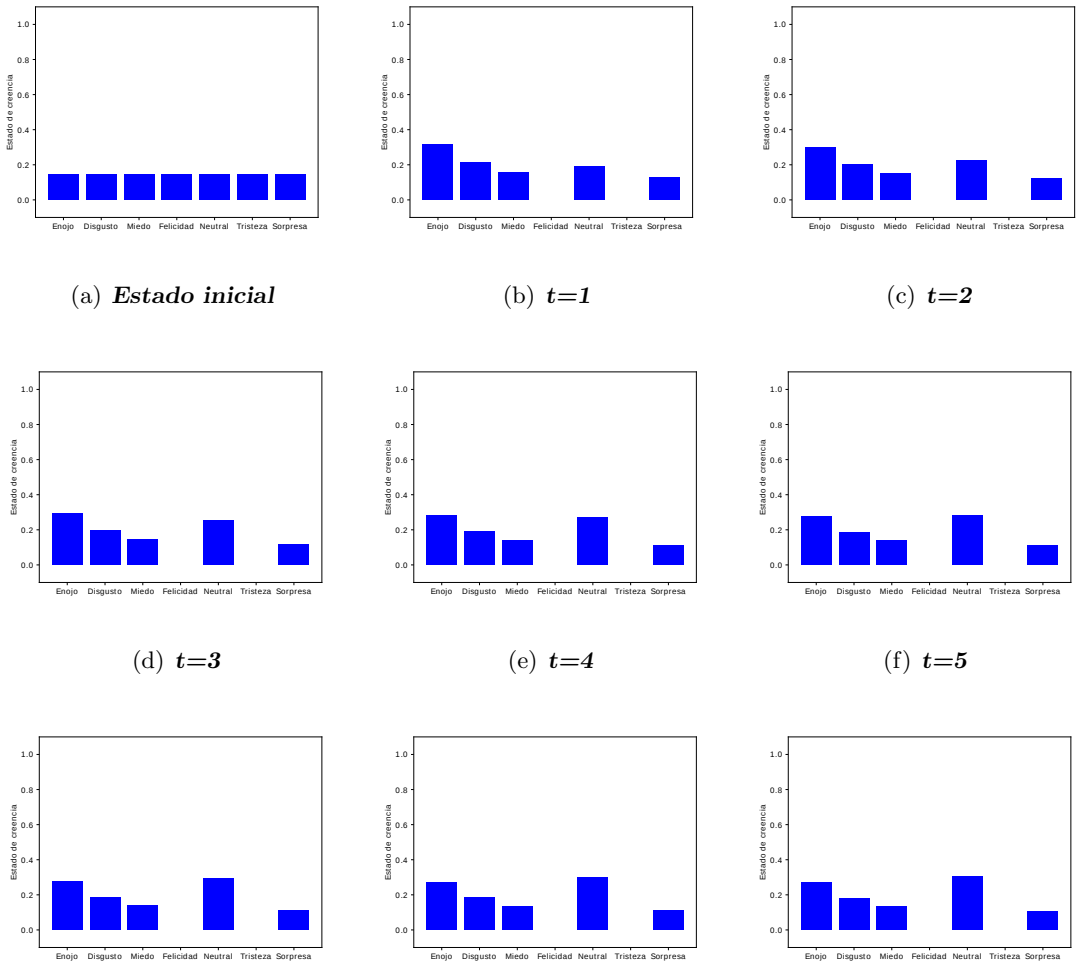
Tabla 4.2: Tiempo en segundos de la predicción emocional

	Horizonte	Enumeración	Tala incremental	Dos pasadas	Testigo
a)	1	0.00,2	0.00,2	0.00,2	0.00,2
	3	0.01,7	0.01,7	0.06,7	0.01,7
	5	0.01,14	0.02,14	0.08,14	0.01,14
	10	0.03,7	0.03,7	0.10,7	0.03,7
	15	0.05,7	0.04,7	0.17,7	0.04,7
b)	32	0.12,6	0.08,6	0.38,6	0.08,6
	64	0.18,6	0.16,6	0.54,6	0.14,6
	128	0.30,6	0.29,6	0.89,6	0.27,6
	256	0.55,6	0.61,6	1.59,6	0.56,6
c)	Método		Horizonte	⟨Tiempo, N. Políticas⟩	error
	Enumeración		490	0.99,6	2.56e-11
	Tala incremental		490	1.15,6	2.56e-11
	Dos pasadas		490	2.92,6	2.56e-11
	Testigo		490	1.05,6	2.56e-11

Tabla 4.3: Tiempo en segundos con la optimización Monte Carlo

	Horizonte	Tiempo (segundos)
a)	1	0.00039 - 0.00079
	3	0.00052 - 0.00102
	5	0.00178 - 0.00379
	10	0.00501 - 0.01376
	15	0.01067 - 0.01715
b)	32	0.01693 - 0.05714
	64	0.03830 - 0.10968
	128	0.13757 - 0.17699
	256	0.17978 - 0.27787
	490	0.42491 - 0.65659

precisión. Aunque éste método es eficiente en velocidad, no es una buena opción por dos razones: Primero, para tener una aproximación más exacta, se requieren varias simulaciones, lo cual afecta al tiempo. Segundo, en nuestro problema no se requiere una planificación tan larga y se ha mostrado que los algoritmos exactos ofrecen un buen rendimiento.

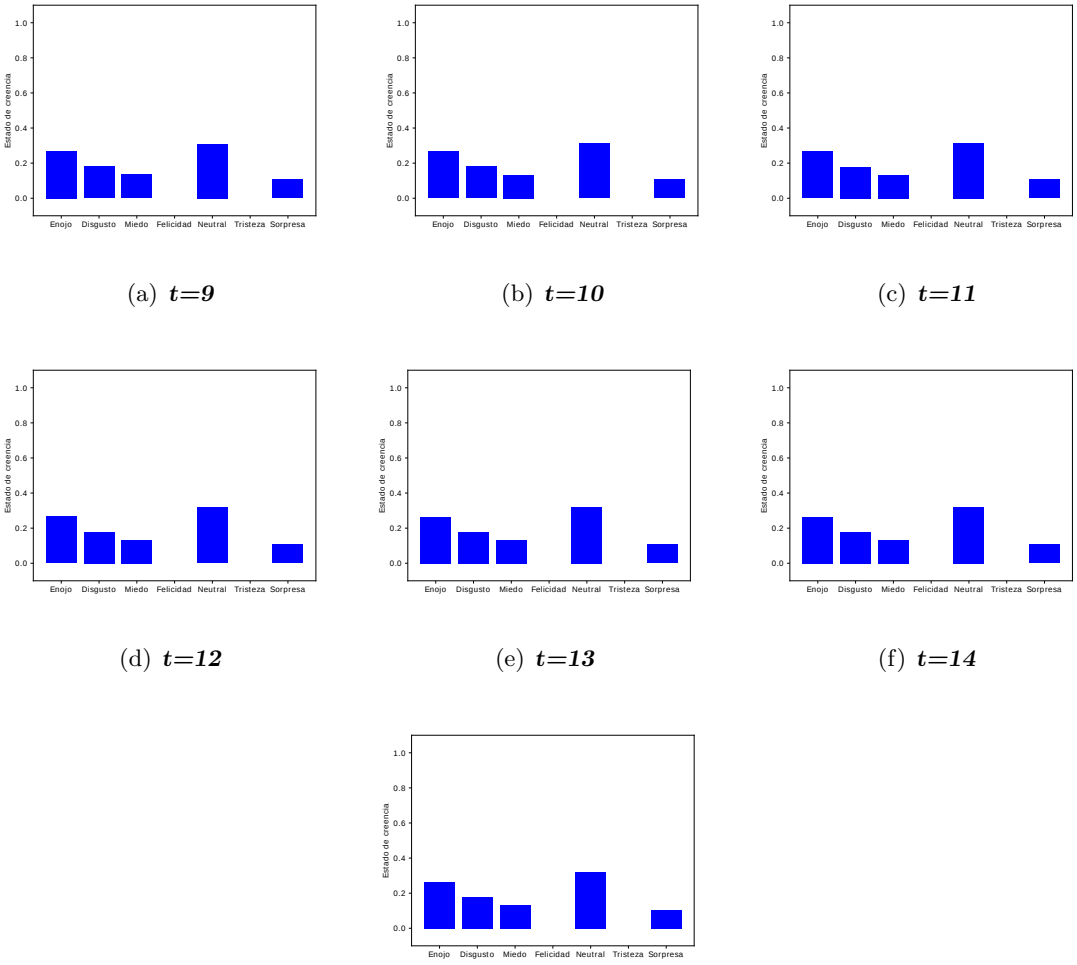


(a) ***Estado inicial*** (b) ***t=1*** (c) ***t=2***

(d) ***t=3*** (e) ***t=4*** (f) ***t=5***

(g) ***t=6*** (h) ***t=7*** (i) ***t=8***

Figura 4.4: Monitoreo del estado emocional de una persona enojada (1/2)



(a) ***t=9*** (b) ***t=10*** (c) ***t=11***

(d) ***t=12*** (e) ***t=13*** (f) ***t=14***

(g) ***t=15***

Figura 4.5: Monitoreo del estado emocional de una persona enojada (2/2)

En este trabajo de tesis lo que se interesa al inicio es que el parlante pueda llegar a un estado neutral. Porqué el estado neutral es una emoción que se relaciona a no percibir tonos altos o bajos de energía, es decir, el parlante debe de estar en un estado de tranquilidad. Todas los parlantes pueden experimentar ésta emoción siempre y cuando el mundo les proporcione las condiciones adecuadas para que se sientan estables.

Las Figuras 4.4 y 4.5 muestran un ejemplo para el monitoreo de emociones de un parlante el cual experimenta constantemente el estado de enojado y el agente trata de llevarlo a un estado neutral dando respuesta a sus comentarios de la mejor manera posible. Ésta tarea le toma al agente entre 5-7 pasos adelante aproximadamente llegar a un estado neutro. En el transcurso del tiempo de estas figuras se muestran las emociones predominantes. Para las demás emociones al agente le toma entre 2-3 pasos adelante aproximadamente realizar éste mismo cambio, a excepción del estado de felicidad y neutral. Esto se debe a que pertenecer a un estado de felicidad lo ideal es mantenerse en el mismo para el caso de un individuo, en el caso de un predictor esto no es aplicable. Por otra parte en el estado neutral se muestra una distribución uniforme entre todos los estados emocionales. Todas estas características dependen del entrenamiento del modelo. Lo importante aquí, es el hecho de que este método llega a la convergencia con una buena velocidad. Obviamente para mejorar éste esquema de predicción, más conversaciones afectivas son requeridas.

En un caso práctico se suponen dos personas las cuales van a realizar un negocio, y un centro de llamadas inteligentes esta monitoreando esa conversación. El centro de llamadas inteligentes no se interesa específicamente cuando alguno llegue a un estado de felicidad, en su lugar es detectar si el negocio resulto bueno para ambas partes. Usualmente en este tipo de actividades, al inicio en ambas partes se pueden percibir valores altos respecto a energía, porqué cada parte busca un punto donde le conviene cerrar un trato. El centro de llamadas inteligentes puede suponer que el negocio fue un éxito si se percibe cualquier emoción positiva o un estado neutral. Si en su lugar se detectan emociones como enojo, disgusto, miedo quiere decir que el negocio no le conviene a alguna parte, porqué alguno tuvo una mayor imponencia.

4.5. Comentarios finales

Existe una amplia cantidad de herramientas para interactuar con el problema de la predicción de emociones. Estas herramientas pueden ser exactas o aproximadas y se basan principalmente en los Procesos de Decisión de Markov con una técnica de optimización para generar y podar políticas. En este Capítulo se exploraron varias estrategias que existen en la literatura para actualizar el estado de creencia. Acerca de los algoritmos implementados, se muestra que los algoritmos exactos proporcionan un buen rendimiento en la predicción. Todas estas herramientas en conjunto permiten desarrollar un sistema en tiempo real para el reconocimiento de las emociones a través de la voz, que tiene una buena precisión para los idiomas en alemán e inglés. Es importante resaltar qué durante la predicción del estado emocional, sólo es importante la observación actual, el estado de creencia que representa los niveles emocionales del parlante en ese momento y el conjunto de observaciones, transiciones y recompensas. En el Capítulo 5 se exponen las conclusiones y trabajo futuro que se relaciona a este trabajo de tesis.

Capítulo 5

Conclusiones

En los capítulos anteriores se ha comentado de los métodos utilizados para realizar una predicción del estado emocional. Ésta predicción depende en gran medida de zonas emocionales distribuidas por diferentes parlantes. En este último capítulo, son expuestas las conclusiones y trabajo futuro que se obtuvieron al desarrollar este sistema de predicción del estado emocional.

5.1. Conclusión

Este trabajo de tesis representa un enfoque diferente para la predicción de emociones que depende solo de señales de voz. La eficiencia de este trabajo de tesis se obtiene al haber divido el problema en dos partes, las cuales son: clasificación y predicción del estado emocional. La eficiencia en la clasificación que se obtiene, se relaciona especialmente al procesamiento realizado con las señales de voz, y al esquema de clasificación de dos etapas. La eficiencia de la predicción se relaciona en gran medida a la etapa de la clasificación, ésta etapa influye directamente en las observaciones de los estados emocionales que se obtienen en la interacción humano-computadora. Las pruebas realizadas en la etapa de predicción, brindan la certeza para elegir el mejor algoritmo en la literatura de los procesos de decisión. En ésta etapa de predicción se muestra que el algoritmo del testigo en general es el más rápido en éste problema, sin embargo, cuando se aproxima a la convergencia el desempeño empieza a degradarse.

5.1.1. Conclusiones particulares

Este trabajo de tesis enumera las siguientes conclusiones particulares:

- Una mejor caracterización se relaciona con el pre-procesamiento, donde los filtros ayudan a limpiar el ruido de las señales de voz y las altas frecuencias para obtener las mejores características.

- Teniendo en cuenta el comportamiento de la señal, eliminando el ruido y preservando los mejores valores para representarlo, permitió que el promedio de clasificación aumentara sin tener que incluir un algoritmo selector de características como lo han desarrollado otros autores.

- En este trabajo de tesis, aunque se tienen los mismos dos valores, el Pitch y la entropía espectral, estos son potenciados por el proceso de caracterización. Por lo tanto, con este método de caracterización, fue posible obtener los resultados que superan el promedio de clasificación con respecto a algunas otras obras del estado del arte para el reconocimiento de las emociones.

- El algoritmo POMCP es muy bueno en problemas de gran dimensión, en especial por el tiempo de computo que se requiere. Éste algoritmo tiene algunas desventajas: La primera, es que se requieren de varias simulaciones para estimar el verdadero valor. La segunda se centra en el número de partículas que se deben elegir, porque si son pocas tiene un pobre desempeño y si son demasiadas se tiene sobre entrenamiento. Estos problemas abarcan especialmente la etapa de la calibración, la ventaja de éste algoritmo aproximado es que puede ser aplicado en problemas bastante largos mostrando buen desempeño.

- En algunos problemas al acercarse a la convergencia, la solución más simple es enumerar todas las políticas. Esto se debe a que implementar alguna otra estrategia de optimización puede degradar el rendimiento.

5.2. Trabajo Futuro

El trabajo futuro que se puede desarrollar para complementar este trabajo de tesis es:

1. Extender estos algoritmos para compararlos con otras bases de datos emocionales, en especial con aquellas que son públicas, gratuitas y de uso académico.

2. Añadir al sistema de reconocimiento de emociones el análisis del contexto y detección de gestos emocionales en las áreas de los ojos y boca por medio de imágenes.

3. Extender el análisis de las emociones a través de la voz, a un sistema que identifique los parlantes y detecte los momentos en que cada uno habla.

Apéndice A

Técnicas de Optimización

El apéndice A se dedica para explicar el funcionamiento de los algoritmos implementados para la planificación a largos horizontes (predicción). Los algoritmos que se incluyen en este apéndice A e.g. Valor iteración, algoritmo del Testigo, algoritmo de Poda incremental, algoritmo de Dos pasadas y la optimización de Monte-Carlo aplicada a los POMDP.

A.1. Valor iteración

El algoritmo valor iteración es considerado ineficiente especialmente por dos razones: Primero, porque se requiere una estimación para todo el espacio $T(s, a, s')$, sin embargo, sólo es importante estimar las porciones del espacio de la política óptima actual. Segundo, las características de los estados no son Markovianas, lo que quiere decir, que se introducirán errores en la política en caso de que no se capture con precisión toda la historia relevante para las condiciones de transición [Young et al., 2013]. El comportamiento de éste algoritmo valor iteración ésta descrito por la ecuación A.1, la cual representa una función híbrida.

$$Q_t^a(s) = R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V_{t-1}(s') \tag{A.1}$$

El Algoritmo 1 describe todos los pasos para el computo de la secuencia de V_t de descuento del valor de las funciones óptimas de horizonte finito, esto se muestra en la

línea 6. En ésta línea 6 también se muestra que se requiere el valor del paso t en el estado s, tomando la acción a, además de la política optima no estacionaria de paso t-1. Éste algoritmo termina cuando la diferencia entre dos sucesivas funciones valor sea menor que algún valor de error ϵ [Kaelbling et al., 1998].

Algoritmo 1 Valor iteración para los MDP

Descripción: Éste algoritmo se encarga de computar la secuencia de V_t de descuento del valor de las funciones óptimas de horizonte finito.

Entra: El conjunto de estados, acciones y observaciones.

Regresa: $V_t(s)$

1: $V_1(s) \leftarrow 0$ para todos estados $s \in S$
2: $t \leftarrow 1$
3: **mientras** $|V_t(s) - V_{t-1}(s)| \geq \epsilon$ para todos los $s \in S$ **hacer**
4: **para cada** $s \in S$ **hacer**
5: **para cada** $a \in A$ **hacer**
6: $Q_t^a(s) \leftarrow R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V_{t-1}(s')$
7: **fin para**
8: $V_t(s) \leftarrow \max_a Q_t^a(s)$
9: **fin para**
10: **fin mientras**

A.2. El algoritmo de Testigo (The Witness Algorithm)

El algoritmo del testigo comienza de manera similar al algoritmo de Cheng con un punto de creencia arbitrario y genera una política. Ésta política se agrega al conjunto de políticas que formaran parte de la función valor final, luego se trata de probar o refutar si éste conjunto realmente corresponde a la función valor final. Al construir una política se realiza una elección para cada observación y de ésta manera obtener una de las políticas de la función valor que representa una estrategia futura particular. Luego, el algoritmo analiza las elecciones individuales una observación a la vez para averiguar que opción daría una mejor función valor. Al igual que los otros algoritmos se define la región donde se está seguro de

que la elección particular es la mejor. Si se puede encontrar un punto de creencia en el que una estrategia diferente sería mejor, esto sirve como un testimonio de que el conjunto actual de políticas aún no es la función valor final real.

En la etapa de actualizar el estado de creencia se utiliza la ecuación A.2; donde b'_o es el estado de creencia que se obtuvo después de haber tomado la acción previa a, una observación actual o, desde el estado de creencia previo b. El valor V de ésta ecuación es el valor $V_t(b) = \max_a Q_t^a(b)$. En los POMDP se tiene un estimador de estado, que en adición con la función de valor $V_t(b)$ de paso-t, se puede definir la expresión $Q_t^a(b)$. Ésta expresión corresponde al valor que se obtiene después de tomar la acción a en el estado de creencia b y continuar de manera óptima desde el paso-(t-1) [Littman, 1994, Cassandra, 1998, Kaelbling et al., 1998].

$$Q_t^a(b) = \sum_{s \in S} b(s) R(s,a) + \gamma \sum_{o \in O} P(o|a,b) V_{t-1}(b'_o) \tag{A.2}$$

El conjunto de funciónes-Q que se obtienen son lineales convexas a pedazos, que se pueden representar mediante árboles de políticas, es decir, la colección de políticas desde un tiempo t. A partir de estas políticas se puede definir un conjunto único útil mínimo de árboles de políticas para cada función-Q, de ésta manera se obtiene el algoritmo 2 [Kaelbling et al., 1998].

El algoritmo 2 es en esencia un algoritmo para resolver un MDP Parcialmente Observable. A éste algoritmo se agrega un nuevo estado, el cual corresponde a maximizar los conjuntos de árboles de políticas representados por la función V_t como sub-conjuntos de árboles de políticas para representar todas las funciones Q_t^a de acuerdo a $V_t \subseteq \bigcup_a Q_t^a$. La condición de paro de éste algoritmo se muestra en la línea 3, la cual se relaciona a un error ϵ. En éste algoritmo también se puede verificar que las *funciones-Q* son separadas por cada acción en la línea 5. Ahora bien, llamar a los testigos se realiza en la línea 6, con la función `witness`, de ahi es donde se obtienen los árboles de políticas. Eliminar las políticas que no aportan información se realiza en la línea 8.

La tarea del algoritmo testigo es buscar un conjunto mínimo de árboles de políticas para representar Q_t^a para cada acción a. Entonces por cada iteración del algoritmo 2 se busca

Algoritmo 2 Ciclo externo del algoritmo testigo

Descripción: Éste algoritmo se encarga de maximizar los conjuntos de árboles de políticas V_t para obtener las funciones Q_t^a.

Entra: V_t, b, a

Regresa: Q_t^a

1: $V_1 \leftarrow \{\langle 0, 0, \ldots, 0\rangle\}$
2: $t \leftarrow 1$
3: **mientras** $|V_t(b) - V_{t-1}(b)| > \epsilon$ **hacer**
4: $t \leftarrow t + 1$
5: **para cada** a en A **hacer**
6: $Q_t^a \leftarrow \text{witness}(V_{t-1}, a)$
7: **fin para**
8: Podar $\bigcup_a Q_t^a$ para obtener V_t
9: **fin mientras**

algún estado de creencia b para el cual el verdadero valor de Q_t^a es calculado por V_{t-1}. Éste valor debe ser diferente del valor previo $\hat{Q}_t^a(b)$ que fue estimado, utilizando el conjunto U_a, es decir, se debe de cumplir $\hat{Q}_t^a(b) \leq Q_t^a(b)$ por todos los b. En caso de encontrar ese estado de creencia se le llama 'testigo', porque da el hecho de que el conjunto U_a aún no es una representación perfecta de Q_t^a. El conjunto U_a de árboles de políticas se inicializa con un único árbol de políticas con la acción a en la raíz, la cual corresponde a un estado de creencias arbitrario.

El proceso para construir árboles de políticas se realiza mientras se tienen puntos 'testigo'. Entonces, para construir el árbol asociado a cada observación o, es necesario el árbol de políticas paso-(t-1) después de ejecutar la acción a. Si esto sucede, el resultado que se obtiene es un árbol p que está construido con sub-árboles p_o. Posteriormente éste se agrega al nuevo árbol de políticas U_a para mejorar la aproximación [Littman, 1994, Cassandra, 1998, Kaelbling et al., 1998].

Acerca de los puntos de testimonio se puede decir que, si p es un árbol de políticas de paso-t, o_i una observación, y p' un árbol de políticas con paso-(t-1), entonces se puede

definir p_{new} como un árbol de políticas de paso-t que se relaciona a p en su acción y todos sus sub-árboles excepto para la observación o_i, para la cual $o_i(p_{new}) = p'$. Esto hace referencia al teorema del testigo [Littman et al., 1995] el cual menciona que si se tiene una mejora sobre todos los árboles de políticas que se han encontrado hasta ahora, entonces b es un testigo, sujetos a la ecuación A.3.

$$V_{p_{new}}(b) > V_{\tilde{p}}(b) \tag{A.3}$$

El teorema del testigo requiere que se busque una $p \in U_a$, $o \in O$, $p' \in V_{t-1}$ y una $b \in B$ tal que se cumple la condición A.3, donde B es el conjunto de todos los estados de creencia compresos en el espacio de estado. Para esto, se obtienen todas las combinaciones p, o y p' para calcular el árbol de políticas p_{new}, en la búsqueda de todos los estados de creencia b.

En la construcción del árbol de políticas la condición $\tilde{p} \in U_a$, $V_{p_{new}}(b) - V_{\tilde{p}}(b)$ permite seguir el árbol de políticas p_{new} en lugar de $\tilde{p}$ la cual comienza desde b. Ahora el problema es encontrar una b que maximice la recompensa sobre todos los árboles de políticas $\tilde{p}$ que el algoritmo ha encontrado hasta ahora. Para esto en [Kaelbling et al., 1998] se propone el programa LP A.4, en el cual la variable δ es valor mínimo de mejora de p_{new} sobre cualquier árbol de políticas en U_a en b. Las restricciones lineales que integra restringen que δ esté unida a la diferencia. Además, las restricciones del método símplex requiere que el estado de creencia b éste normalizado. Éste algoritmo maximiza la recompensa de p_{new} sobre todo $\tilde{p} \in U_a$. Finalmente, el estado b es un punto testigo cuando se cumple que la mayor recompensa es positiva, es decir $\delta > 0$, lo que también significa que p_{new} es una mejora en todos los $\tilde{p}$ árboles. Para más referencia se puede consultar [Littman, 1994, Cassandra, 1998, Kaelbling et al., 1998]

$$
\begin{aligned}
&\textbf{Entra:} && \mathrm{U}_a,\ p_{new} \\
&\underset{\delta, b(s)}{\textbf{Maximiza:}} && \delta \\
&\text{s.t.} && V_{p_{new}}(b) - V_{\tilde{p}}(b) \geq \delta,\ \tilde{p} \in U_a \\
& && \sum_{s \in S} b(s) = 1 \\
& && \delta, b \geq 0
\end{aligned} \tag{A.4}
$$

A.3. Algoritmo de Poda Incremental (Incremental Pruning Algorithm)

El algoritmo de Poda incremental utiliza una notación vectorial para representar las políticas, y el estado de creencia. Por este motivo antes de explicar el algoritmo completo es conveniente conocer algunas notaciones vectoriales. Se tiene dos vectores α y β con $|S|$ cantidad de estados, entonces, se dice que $\alpha \geq \beta$ si y solo si $\alpha(s) \geq \beta(s)$ por cada estado $s \in S$. La suma de vectores también se realiza elemento por elemento. El producto escalar de vectores se define por $\alpha \cdot \beta = \sum_s \alpha(s)\beta(s)$. El vector '0', es un vector donde todos sus elementos son ceros. De manera similar el vector '1' es un vector donde todos sus elementos son unos. El vector unitario e_s es aquel donde todos sus elementos son ceros excepto $e_s(s) = 1$. El producto suma de dos conjuntos de vectores es igual a $A \oplus B = \{\alpha + \beta | \alpha \in A, \beta \in B\}$. El conjunto substracción ésta definido por $A \backslash B = \{\alpha | \alpha \in A, \alpha \notin B\}$.

Ahora, por parte del algoritmo de poda incremental se puede caracterizar el conjunto de estados S (ecuación A.5) descrito por acciones como en la ecuación A.6 además de acciones y observaciones, de acuerdo a la ecuación A.7.

$$S' = purge(\bigcup_{a \in A} S^a) \tag{A.5}$$

$$S^a = purge(\oplus_{o \in O} S^a_o) \tag{A.6}$$

$$S^a_o = purge(\{\tau(\alpha, a, o) | \alpha \in S\}), \tag{A.7}$$

donde:

$$\tau(\alpha, a, o)(s) = (1/|O|)r^a(s) + \gamma \sum_{s'} \alpha(s')Pr(o|s', a)Pr(s'|s, a) \tag{A.8}$$

La función $purge(\cdot)$ toma un conjunto de vectores de políticas y lo reduce a su única forma mínima, de acciones que se pueden tomar de acuerdo a:

$$purge(A) = \{\alpha|\alpha \in A, R_{testigo}(\alpha, A) \neq \emptyset\} \tag{A.9}$$

$$R_{testigo}(\alpha, A) = \{b|b \geq 0, b \cdot 1 = 1, b \cdot \alpha > b \cdot \alpha', \alpha' \in A \backslash \{\alpha\}\} \tag{A.10}$$

El conjunto $R_{testigo}(\alpha, A)$ es conocido como la región testigo del vector α, el cual verifica si el vector α es necesario para representar la función híbrida de acuerdo a $A \bigcup \{\alpha\}$ y b corresponde a la distribución de probabilidad sobre los estados posibles ($b(s)$ es la probabilidad de que el agente se encuentra en el estado s).

El funcionamiento del algoritmo de poda incremental se muestra con los Algoritmos 3, 4 y 5. En estos algoritmos la variable W representa el conjunto de vectores ganadores, es decir, aquellos que tienen las regiones de los testigos no vacías.

Algoritmo 3 Poda incremental: PRINCIPAL

Descripción: Ésta es la rutina principal, la cual recibe el conjuntos estados, acciones y observaciones representador por vectores.

Entra: $S^a_{o1}, S^a_{o2}, \ldots, S^a_{ok}$

Regresa: W **el cual corresponde al vector de políticas**

1: $W \leftarrow$ FILTRO$(S^a_{o1} \oplus S^a_{o2})$
2: **para** $i \leftarrow 3$ **hasta** k **hacer**
3: $\quad W \leftarrow$ FILTRO $(W \oplus S^a_{oi})$
4: **fin para**
5: **devolver** W

Algoritmo 4 Poda incremental: FILTRO

Descripción: Ésta rutina construye la función híbrida por pasos, dado el conjunto de estados, acciones y observaciones. Además se encarga de eliminar todos los vectores de políticas que no aportan información utilizando el Programa LP.

Entra: F el cual representa la combinación de vectores de políticas de la linea 1 y 3 del Algoritmo 3.

Regresa: W

1: $W \leftarrow \emptyset$
2: **para cada** s en S **hacer**
3: $\omega \leftarrow \arg\max_{\phi \in F} e_s \cdot \phi$
4: $W \leftarrow W \bigcup \{\omega\}$
5: $F \leftarrow F \backslash \{\omega\}$
6: **mientras** $F \neq \emptyset$ **hacer**
7: $b \leftarrow$ ProgramaLP(ϕ, W)
8: **si** $b = \perp$ **entonces**
9: $F \leftarrow$ F $\backslash \{\phi\}$
10: **si no**
11: $\omega \leftarrow \arg\max_{\phi \in F} x \cdot \phi$
12: $W \leftarrow W \bigcup \{\omega\}$
13: $F \leftarrow F \backslash \{\omega\}$
14: **fin si**
15: **fin mientras**
16: **fin para**
17: **devolver** W

Algoritmo 5 Poda incremental: ProgramaLP

Descripción: Ésta es una rutina de programación lineal que se utiliza para resolver el algoritmo de poda incremental por la mejor acción.

Entra: α, A

Regresa: El vector b representa el estado de creencia.

```
1: L ← LP(variables:b(s); objetivo: max δ)
2: para cada α' en A\{α} hacer
3:    AgregarRestricción(L, b•1 = 1)
4:    AgregarRestricción(L, b ≥0)
5:    si INFACTIBLE(L) entonces
6:       devolver ⊥
7:    si no
8:       (b, δ) ← ResolverLP(L)
9:       si δ > 0 entonces
10:         devolver (b)
11:      si no
12:         devolver ⊥
13:      fin si
14:   fin si
15: fin para
```

A.4. Algoritmo de Dos Pasadas (The two pass algorithm)

Para ilustrar éste algoritmo se va a representar la función valor de acuerdo a la siguiente notación $\Gamma_n^{a,o}$, la cual representa el conjunto de vectores de políticas Γ_n para n=0 , ..., N-1, que forman la función híbrida, por acciones y observaciones. El algoritmo de dos pasadas podría determinar si un vector que se obtiene de dos vectores $\alpha + \beta$ es útil. Ésta tarea se logra al determinar si está vacía la región testigo $R_{testigo}$ definida por $R_{testigo}(\alpha,\text{A}) \bigcap R_{testigo}(\beta,\text{B})$. Ahora bien, $\gamma_n^{a,o}$ son todos los n posibles vectores de políticas útiles dada una acción a y una observación o. El conjunto de vectores que se obtienen al

seleccionar alguna observación o de todas las maneras posibles ésta dado por:

$$\gamma_n^a = \sum_o \gamma_n^{a,o} = \gamma_n^{a,0} + \gamma_n^{a,1} + \gamma_n^{a,2} + \dots \tag{A.11}$$

Para comprobar si el conjunto de vectores γ_n^a es útil y suficiente, basta por comprobar la región $R_{testigo}$ si se encuentra vacía o no de acuerdo a:

$$\bigcap_o R_{testigo}(\gamma_n^{a,o}, \Gamma_n^{a,o}) = R_{testigo}(\gamma_n^{a,0}, \Gamma_n^{a,0}) \bigcap R_{testigo}(\gamma_n^{a,1}, \Gamma_n^{a,1}) \bigcap R_{testigo}(\gamma_n^{a,2}, \Gamma_n^{a,2}) \bigcap \dots \tag{A.12}$$

La región actual que se define por Γ_n^a puede ser simplificada por:

$$\bigcap_o R_{testigo}(\gamma_n^{a,o}, \Gamma_n^{a,o}) = R_{testigo}(\gamma_n^a, \Gamma_n^a) \tag{A.13}$$

El algoritmo 6 muestra los pasos para obtener Γ_n^a usando el algoritmo de dos pasadas. En éste algoritmo $\hat{\Gamma}$ contiene los vectores de políticas que se han encontrado y cuyas regiones ya se han explorado, Υ contiene los vectores de políticas que se han encontrado cuyas regiones aún no se han explorado. La implementación de éste algoritmo requiere mantener la información sobre cómo los vectores γ_n^a fueron construidos, es decir cómo los vectores $\gamma_n^{a,o}$ fueron utilizados.

En el algoritmo 6 el primer ciclo que se muestra en la línea 4 trabaja sobre los vectores de políticas cuyas regiones aún no se han explorado. El segundo ciclo se muestra en la línea 8, el cual es responsable de explorar por cada región los posibles vectores de políticas vecinos ($\mathcal{N}$) que pueden formar alguno de los bordes. Si algún vector vecino forma un borde, éste se agrega a Υ para una exploración más tarde.

El algoritmo de dos pasadas explora una región usando un programa de programación lineal el cual se muestra en la línea 9 con la instrucción 'ProgramaLP'. Éste programa LP es similar a los comentados anteriormente. El problema LP en éste algoritmo tiene las variables del estado de creencia $b(s)$. La función objetivo es $b_{\blacksquare}(\nu\text{-}\gamma_n^a)$ con las restricciones $b \in R_{testigo}$, es decir, que el estado de creencia del vector que se obtiene pertenece a la región del testigo, y $b \in \Pi(\mathcal{S})$ que corresponde a la distribución de probabilidad sobre el conjunto

de estados. Si un vector vecino define un borde, entonces éste LP debería ser factible con puntos de solución a lo largo del borde.

El principal problema del algoritmo de dos pasadas es que no se tiene garantía de obtener el conjunto parsimonioso, sin embargo, es necesario incluir estos vectores de políticas adicionales que son mejor conocidos como 'vecinos impostores' debido a que todas las regiones adyacentes están formadas por vecinos. Para finalizar, existen algunas estrategias para evitar crear estos 'vecinos impostores' pero existe la posibilidad de caer en un problema de estar más allá de la frontera de la región, esto quiere decir de una manera simple, que estos 'vecinos impostores' son necesarios.

Algoritmo 6 Dos pasadas

Descripción: Éste algoritmo muestra el procedimiento para la construcción de Γ_n^a, el cual representa la función híbrida.

Entra: Γ_{n-1}, a

Regresa: $\hat{\Gamma}$

1: $\hat{\Gamma} := \emptyset$
2: $b =$ alguna información del estado
3: $\Upsilon = \{\gamma_n^a(b)\}$
4: **mientras** $\Upsilon \neq \emptyset$ **hacer**
5: $\quad \gamma_n^a =$ remover un vector de Υ
6: $\quad \hat{\Gamma} = \hat{\Gamma} \bigcup \{\gamma_n^a\}$
7: $\quad \hat{R} = \bigcap_o R_{testigo}(\gamma_n^{a,o}, \Gamma_n^{a,o})$
8: $\quad$ **para cada** $\nu \in \mathcal{N}(\gamma_n^a)$ **hacer**
9: $\quad\quad L =$ ProgramaLP$(\gamma_n^a, \nu, \hat{R})$
10: $\quad\quad$ **si** FACTIBLE(L) y $\nu \notin \hat{\Gamma}$ **entonces**
11: $\quad\quad\quad \Upsilon = \nu \bigcup \{\upsilon\}$
12: $\quad\quad$ **fin si**
13: $\quad$ **fin para**
14: **fin mientras**
15: **devolver** $\hat{\Gamma}$

A.5. Optimización Monte-Carlo

El método POMCP combina la actualización del estado de creencia del agente (POMDP) con el árbol de búsqueda Monte-Carlo (Monte-Carlo Tree Search, por sus siglas del inglés) del actual estado de creencia. El valor del nodo de una historia h del árbol de búsqueda es estimado alcanzando la convergencia cada vez que se obtiene un estado de creencia correcto [Silver and Veness, 2010].

El valor de la *función-Q* en éste algoritmo se estima de manera *online*. Ésta estimación guía la selección de futuras acciones, y su efecto se ve reflejado en un menor tiempo de exploración. Las actualizaciones se realizan al final de cada simulación, esto significa, que las recompensas de todas las simulaciones pueden ser utilizadas para valorar directamente la política actual. Ésta tarea disminuye los efectos de cualquier dinámica no Markoviana en las características del estado [Young et al., 2013].

El algoritmo POMCP tiene dos características sobresalientes: La primera, se relaciona con romper los efectos adversos de la dimensionalidad durante la actualización del estado de creencia y la planificación, en ésta parte es donde *Monte-Carlo sampling* tiene lugar. La segunda propiedad, se relaciona a un simulador de caja negra para problemas que son grandes o complejos para ser representados [Silver and Veness, 2010].

El agente utiliza un modelo generativo G como simulador del POMDP. Éste agente G ésta dado por $G(s_t, a_t) \leftarrow (s_{t+1}, o_{t+1}, r_{t+1})$ que puede ser reiniciado al inicio del estado s [Silver and Veness, 2010]. El valor de regreso con descuento de cada punto del estado de creencia visitado en el tiempo t es dado por la ecuación A.14 donde γ es valor de descuento [Young et al., 2013].

$$R_{i(t)} = \sum_{t \leq \tau \leq T} \gamma^{(\tau - t)} r_\tau \tag{A.14}$$

Un método simple para evaluar un estado s es utilizando una política aleatoria *rollout*. Ésta política es utilizada hasta que sea alcanzado algún estado o horizonte con descuento. Si tenemos N simulaciones, entonces se puede estimar el valor del estado por el promedio de los valores de las R_i simulaciones, es decir $V(s) = 1/N \sum_{i=1}^{N} R_i$. Evaluar un POMDP con una acción a en la historia h y la política *rollout* se representa por la política

$\pi_{rollout}(h, a)$ [Silver and Veness, 2010].

En cada nodo del árbol se tiene que por cada estado s y acción a contiene un valor *Q(s, a)* y un número de visitas *N(s, a)*. El valor se estima por el promedio de las simulaciones en las cuales se toman todas las acciones a de algún estado s. El número de visitas es la cantidad de veces que se accede a ese estado. De la notación anterior se puede apreciar que el número de visitas total por algún estado s equivale a $N(s) = \sum_a N(s, a)$. Al iniciar el algoritmo, el valor y el número de visitas equivale a cero, es decir, $Q(s, a) = 0$, $N(s, a) = 0$ [Silver and Veness, 2010]. La historia h representa el árbol de búsqueda de Monte-Carlo. Al final de cada simulación, se aplican las siguientes ecuaciones para actualizar cada nodo de árbol de búsquedas:

$$\hat{Q}(s, a) = Q(s, a) + R_i \tag{A.15}$$

$$\hat{N}(s, a) = N(s, a) + 1 \tag{A.16}$$

para $t = 0, 1, \ldots, T$. Los valores-Q pueden entonces ser re-estimados para todos los puntos de creencia i visitados en la simulación y todos los a como:

$$\hat{Q}(s, a) = \frac{1}{N(s, a)} Q(s, a) \tag{A.17}$$

Para converger a una política óptima en la simulación del sistema, se debería tomar una mezcla del óptimo actual y las acciones exploradas:

$$a = \begin{cases} \text{Acción aleatoria} & \text{con probabilidad} \quad \epsilon \\ \arg\max_a Q(s, a) & \text{con probabilidad} \quad 1 - \epsilon \end{cases} \tag{A.18}$$

donde ϵ controla la tasa de exploración. Típicamente esto es grande inicialmente, pero se reduce a medida que el aprendizaje progresa [Young et al., 2013]. Para seleccionar las acciones en MCTS de una mejor manera, existe un algoritmo llamado [Méhat and Cazenave, 2010, Silver and Veness, 2010] en el cual el valor de las acciones se adquiere por un bono de exploración.

$$\hat{Q}(s, a) = Q(s, a) + c\sqrt{\frac{\log N(s)}{N(s, a)}} \tag{A.19}$$

En la ecuación A.19 el escalar c determina la proporción relativa de exploración. Si $c = 0$ entonces el algoritmo funciona de una manera voraz dentro del árbol de búsquedas. En ésta ecuación se requiere la acción que maximiza el valor del estado s lo que equivale a $argmax_a Q(s, a)$ [Silver and Veness, 2010].

Ahora se va a utilizar el concepto de una historia h para describir los nodos del árbol de búsquedas. Cada nodo se representa por $T(h) = \langle N(h), V(h) \rangle$, estos nodos pueden ser estimados por el promedio de todas las simulaciones en la historia h. Por lo tanto *N(h)* corresponde al número de veces que la historia h ha sido visitada y de la misma manera *V(h)* es el valor de la historia h [Silver and Veness, 2010].

El autor Speekenbrink [Speekenbrink, 2016] explica en qué consiste el filtro de partículas debido a que en éste método es necesario para actualizar el estado de creencia a través de una aproximación. Los autores Silver y Veness [Silver and Veness, 2010] actualizan el estado de creencia de la historia h_t con K partículas $B_t^i \in S, 1 \leq i \leq K$. Donde cada partícula corresponde a la muestra del estado s y el estado de creencia es la suma de todas las partículas donde δ_s es función delta de kronecker [Silver and Veness, 2010].

$$\hat{B}(s, h_t) = \frac{1}{K} \sum_{i=1}^{K} \delta_{sB_t^i} \tag{A.20}$$

En la practica el filtro de partículas en los POMDP actúa de la siguiente manera: Las partículas son actualizadas después de una acción a_t y obtener una observación o_t. Un estado s es muestreado por el estado de creencia actual $B(s, h_t)$ por la selección de una partícula aleatoria de B_t. Ésta partícula pasa a través de la caja negra para obtener un estado sucesor s' y una observación o' es decir $G(s, a_t) \leftarrow (s', o', r)$. Si la muestra de la observación se relaciona a la observación real $o = o_t$, entonces una nueva partícula s' es añadida a B_{t+1}. Éste proceso se repite hasta que las K partículas han sido agregadas [Silver and Veness, 2010].

Una consideración importante para la optimización de Monte Carlo es el número de partículas a utilizar. Si se utilizan demasiadas partículas el rendimiento será deficiente resultará debido al exceso del entrenamiento. A la inversa, tener muy pocas conduce a un rendimiento deficiente debido a la falta de discriminación en la toma de decisiones

[Young et al., 2013]. Finalmente, el POMCP se muestra con los algoritmos 7, 8 y 9.

Algoritmo 7 POMCP: SIMULACIÓN

Descripción: Ésta rutina es la encargada del funcionamiento del algoritmo POMCP en general, entre sus funciones se encuentra crear, y actualizar las ramas del árbol de Monte Carlo. En ésta rutina se puede observar en la linea 12 una variable que se llama *hao*, la cual representa la historia h, acción a y una observación o del árbol de búsquedas.

Entra: s**,** h**,** *depth*

Regresa: R

1: **si** $\gamma^{depth} < \epsilon$ **entonces**
2: **devolver** 0
3: **fin si**
4: **si** $h \notin T$ **entonces**
5: **para cada** $a \in A$ **hacer**
6: $T(ha) \leftarrow (N_{init}(ha), V_{init}(ha), \emptyset)$
7: **fin para**
8: **devolver** ROLLOUT(s, h, *depth*)
9: **fin si**
10: $a \leftarrow \arg\max_b V(hb) + c\sqrt{\frac{\log N(h)}{N(hb)}}$
11: (s',o, r) $\sim G(s, a)$
12: $R \leftarrow r$ + SIMULACIÓN(s',*hao*, *depth* + 1)
13: $B(h) \leftarrow B(h) \bigcup \{\text{s}\}$
14: $N(h) \leftarrow N(h) + 1$
15: $N(ha) \leftarrow N(ha) + 1$
16: $V(ha) \leftarrow V(ha) + \frac{R - V(ha)}{N(ha)}$
17: **devolver** R

Algoritmo 8 POMCP: BÚSQUEDA

Descripción: Ésta rutina es la encargada de ejecutar un número de simulaciones o utilizar tiempo máximo de ejecución para entregar la estimación del estado.

Entra: La historia h

Regresa: $\arg\max_b V(hb)$

mientras TIMEOUT() $\neq$ True **hacer**
  si $h =$ *vacía* **entonces**
    $s =$ Distribución uniforme sobre el estado de creencia
  si no
    $s \sim B(h)$
  fin si
  SIMULACIÓN(s, h, 0)
fin mientras
devolver $\arg\max_b V(hb)$

Algoritmo 9 POMCP: ROLLOUT

Descripción: Ésta rutina obtiene la política aleatoria *rollout* utilizando la caja negra que corresponde al generador y al filtro de partículas.

Entra: s, h, $depth$

Regresa: **r** $+ \gamma$ **ROLLOUT(**s', hao, $depth + 1$**)**

si $\gamma^{depth} < \epsilon$ **entonces**
  devolver 0
fin si
$a \sim \pi_{rollout}(h, \cdot)$
$(s', o, r) \sim G(s, a)$
devolver r + γ ROLLOUT(s', hao, $depth + 1$)

Referencias

[Badshah et al., 2016] Badshah, A. M., Ahmad, J., Lee, M. Y., and Baik, S. W. (2016). Divide-and-conquer based ensemble to spot emotions in speech using mfcc and random forest. *arXiv preprint arXiv:1610.01382*.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[Cassandra, 1998] Cassandra, A. R. (1998). Exact and approximate algorithms for partially observable markov decision processes.

[Chong et al., 2008] Chong, E. K., Kreucher, C. M., and Hero, A. O. (2008). Monte-carlo-based partially observable markov decision process approximations for adaptive sensing. In *Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on*, pages 173–180. IEEE.

[Christopher, 2016] Christopher, M. B. (2016). *PATTERN RECOGNITION AND MACHINE LEARNING.* Springer-Verlag New York.

[Darekar and Dhande, 2017] Darekar, R. and Dhande, A. (2017). Toward improved performance of emotion detection: Multimodal approach. In *Proceedings of the International Conference on Data Engineering and Communication Technology*, pages 431–443. Springer.

[Duda et al., 2012] Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.

[El Ayadi et al., 2011] El Ayadi, M., Kamel, M. S., and Karray, F. (2011). Survey on speech

emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3):572–587.

[Fadil et al., 2015] Fadil, C., Alvarez, R., Martínez, C., Goddard, J., and Rufiner, H. (2015). Multimodal emotion recognition using deep networks. In *VI Latin American Congress on Biomedical Engineering CLAIB 2014, Paraná, Argentina 29, 30 & 31 October 2014*, pages 813–816. Springer.

[Gerhard, 2003] Gerhard, D. (2003). *Pitch extraction and fundamental frequency: History and current techniques*. Department of Computer Science, University of Regina Regina.

[Geurts et al., 2006] Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42.

[Grimm et al., 2007] Grimm, M., Kroschel, K., Harris, H., Nass, C., Schuller, B., Rigoll, G., and Moosmayr, T. (2007). On the necessity and feasibility of detecting a driver?s emotional state while driving. *Affective computing and intelligent interaction*, pages 126–138.

[Gu et al., 2016] Gu, Y., Postma, E. O., Lin, H.-X., and van den Herik, H. J. (2016). Speech emotion recognition with log-gabor filters. In *ICAART (2)*, pages 446–452.

[Hale, 2017] Hale, L. T. (2017). Affective speech design: Emotional i/o. In *Advances in Affective and Pleasurable Design*, pages 377–383. Springer.

[Hoey and Schröder, 2015] Hoey, J. and Schröder, T. (2015). Bayesian affect control theory of self. In *AAAI*, pages 529–536.

[Hoey et al., 2016] Hoey, J., Schröder, T., and Alhothali, A. (2016). Affect control processes: Intelligent affective interaction using a partially observable markov decision process. *Artificial Intelligence*, 230:134–172.

[Isaacson, 2013] Isaacson, L. K. (2013). Spectral entropy, empirical entropy and empirical exergy for deterministic boundary-layer structures. *Entropy*, 15(10):4134–4158.

[Kaelbling et al., 1998] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134.

[Kyriakos,] Kyriakos, P. Planning in pomdps using mdp heuristics.

[Li et al., 2012] Li, J., Zhou, P., Jing, X., and Du, Z. (2012). Speech endpoint detection method based on teo in noisy environment. *Procedia Engineering*, 29:2655–2660.

[Littman, 1994] Littman, M. L. (1994). The witness algorithm: Solving partially observable markov decision processes. *Brown University, Providence, RI.*

[Littman et al., 1995] Littman, M. L., Cassandra, A. R., and Kaelbling, L. P. (1995). Efficient dynamic-programming updates in partially observable markov decision processes.

[Mattheyses et al., 2006] Mattheyses, W., Verhelst, W., and Verhoeve, P. (2006). Robust pitch marking for prosodic modification of speech using td-psola. In *Proceedings of the IEEE Benelux/DSP Valley Signal Processing Symposium, SPS-DARTS*, pages 43–46.

[Méhat and Cazenave, 2010] Méhat, J. and Cazenave, T. (2010). Combining uct and nested monte carlo search for single-player general game playing. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(4):271–277.

[Monahan, 1982] Monahan, G. E. (1982). State of the art-a survey of partially observable markov decision processes: theory, models, and algorithms. *Management Science*, 28(1):1–16.

[Norouzi et al., 2015] Norouzi, M., Collins, M., Johnson, M. A., Fleet, D. J., and Kohli, P. (2015). Efficient non-greedy optimization of decision trees. In *Advances in Neural Information Processing Systems*, pages 1729–1737.

[Pohjalainen and Alku, 2014] Pohjalainen, J. and Alku, P. (2014). Multi-scale modulation filtering in automatic detection of emotions in telephone speech. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 980–984. IEEE.

[Poria et al., 2016a] Poria, S., Cambria, E., Howard, N., Huang, G.-B., and Hussain, A. (2016a). Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomputing*, 174:50–59.

[Poria et al., 2016b] Poria, S., Chaturvedi, I., Cambria, E., and Hussain, A. (2016b). Convolutional mkl based multimodal emotion recognition and sentiment analysis. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 439–448. IEEE.

[Ramakrishnan and El Emary, 2013] Ramakrishnan, S. and El Emary, I. M. (2013). Speech emotion recognition approaches in human computer interaction. *Telecommunication Systems*, pages 1–12.

[Ridgeway, 2007] Ridgeway, G. (2007). Generalized boosted models: A guide to the gbm package. *Update*, 1(1):2007.

[Rybka and Janicki, 2013] Rybka, J. and Janicki, A. (2013). Comparison of speaker dependent and speaker independent emotion recognition. *International Journal of Applied Mathematics and Computer Science*, 23(4):797–808.

[Schölkopf, 2001] Schölkopf, B. (2001). The kernel trick for distances. In *Advances in neural information processing systems*, pages 301–307.

[Shawe-Taylor and Cristianini, 2004] Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

[Silver and Veness, 2010] Silver, D. and Veness, J. (2010). Monte-carlo planning in large pomdps. In *Advances in neural information processing systems*, pages 2164–2172.

[Speekenbrink, 2016] Speekenbrink, M. (2016). A tutorial on particle filters. *Journal of Mathematical Psychology*, 73:140–152.

[Suykens et al., 2003] Suykens, J. A., Van Gestel, T., Vandewalle, J., and De Moor, B. (2003). A support vector machine formulation to pca analysis and its kernel version. *IEEE Transactions on neural networks*, 14(2):447–450.

[Thrun, 2000] Thrun, S. (2000). Monte carlo pomdps. In *Advances in neural information processing systems*, pages 1064–1070.

[Ververidis and Kotropoulos, 2003] Ververidis, D. and Kotropoulos, C. (2003). A review of emotional speech databases. In *Proc. Panhellenic Conference on Informatics (PCI)*, pages 560–574.

[Ververidis et al., 2004] Ververidis, D., Kotropoulos, C., and Pitas, I. (2004). Automatic emotional speech classification. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–593. IEEE.

[Wagner et al., 2007] Wagner, J., Vogt, T., and André, E. (2007). A systematic comparison of different hmm designs for emotion recognition from acted and spontaneous speech. In *International Conference on Affective Computing and Intelligent Interaction*, pages 114–125. Springer.

[Williams and Young, 2007] Williams, J. D. and Young, S. (2007). Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422.

[Yaacob et al., 2015] Yaacob, S., Muthusamy, H., and Polat, K. (2015). Improved emotion recognition using gaussian mixture model and extreme learning machine in speech and glottal signals.

[Yaacob and Polat, 2017] Yaacob, S. and Polat, K. (2017). Bispectral features and mean shift clustering for stress and emotion recognition from natural speech r.

[Yogesh et al., 2017] Yogesh, C., Hariharan, M., Ngadiran, R., Adom, A. H., Yaacob, S., Berkai, C., and Polat, K. (2017). A new hybrid pso assisted biogeography-based optimization for emotion and stress recognition from speech signal. *Expert Systems with Applications*, 69:149–158.

[Young et al., 2010] Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The hidden information state model: A practical framework

for pomdp-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

[Young et al., 2013] Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

[Zhang et al., 2006] Zhang, H., Berg, A. C., Maire, M., and Malik, J. (2006). Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2126–2136. IEEE.

[Zhang et al., 2017] Zhang, W., Zhao, D., Chai, Z., Yang, L. T., Liu, X., Gong, F., and Yang, S. (2017). Deep learning and svm-based emotion recognition from chinese speech for smart affective services. *Software: Practice and Experience*.

[Zhou et al., 2010] Zhou, E., Fu, M. C., and Marcus, S. I. (2010). Solving continuous-state pomdps via density projection. *IEEE Transactions on Automatic Control*, 55(5):1101–1116.

[Zhu et al., 2009] Zhu, J., Zou, H., Rosset, S., Hastie, T., et al. (2009). Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360.