МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет радиофизики и электроники

Кафедра радиофизики

АНАЛИЗ МУЗЫКАЛЬНЫХ ДАННЫХ ПО АЛГОРИТМУ ИЕРАРХИЧЕСКОЙ ВРЕМЕННОЙ ПАМЯТИ

Дипломная работа
студента 5 курса 1 группы
ЛАГОДЫ Дмитрия Александровича

Научный руководитель: ХЕЙДОРОВ И.Э.,
доцент кафедры радиофизики,
кандидат физико-математических наук

Рецензент: ДИГРИС А.В.,
ассистент кафедры системного анализа

«Допустить к защите»
зав. кафедрой радиофизики
д-р ф.–м. н. _____/Рудницкий А.С./
« ___» _____ 2010г.

Минск, 2010

РЕФЕРАТ

Дипломная работа 49 страниц, 16 иллюстраций, 24 источника, 1 приложение.

МУЗЫКА, АНАЛИЗ ДАННЫХ, МАШИННОЕ ОБУЧЕНИЕ, ИЕРАРХИЧЕСКАЯ ВРЕМЕННАЯ ПАМЯТЬ, ОБРАБОТКА ЗВУКА, МОДЕЛИРОВАНИЕ

Объектом исследования является иерархическая временная память в применении к анализу музыкальных данных. Цель работы – создать систему для автоматического анализа музыкальных коллекций.

Проанализированы пути анализа музыкальных данных. Исследовано автоматическое построение моделей музыкальных данных и использование их в задачах классификации. Проведен сравнительный численный эксперимент идентификации исполнителей и классификации композиций на наличие вокала.

Показано что иерархическая временная память может служить промежуточным звеном между численными музыкальными моделями и алгоритмами классификации для улучшения качества расчетов и ускорения вычислений.

Также показано, что текущие реализации алгоритмов иерархической временной памяти не обладают необходимыми свойствами для достижения значительных результатов в области автоматического создания обобщенных музыкальных моделей, и показаны пути, для решения этих вопросов.

На текущий момент Иерархическая Временная Память может быть эффективно применима как дополнение к анализу и моделированию музыкальных структур.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ ПРИНЯТЫХ ОБОЗНАЧЕНИЙ 4
ВВЕДЕНИЕ 5
ЦЕЛЬ И ЗАДАЧИ 6
1 ПРЕДМЕТНАЯ ОБЛАСТЬ - MIR 7
1.1 Решаемые вопросы 7
1.2 Связь с другими областями науки 7
1.3 Способы описания музыки 8
1.4 Методы получения данных о музыке 8
1.5 Иерархичность музыкальных структур 9
2 ИЕРАРХИЧЕСКАЯ ВРЕМЕННАЯ ПАМЯТЬ 12
2.1 Предисловие 12
2.2 Что делает НТМ 13
2.2.1 Обнаружение причин в мире 14
2.2.2 Выдвижение гипотез о причинах новой информации 17
2.2.3 Предсказания 19
2.3 Обнаружение причин и выдвижение гипотез НТМ 20
2.4 О важности иерархии 24
2.5 Обнаружение причин и выдвижение гипотез отдельными узлами 24
2.6 О необходимости времени для обучения 29
3 МЕТОДЫ ПРЕДОБРАБОТКИ И КЛАССИФИКАЦИИ ДАННЫХ 32
3.1 Предобработка музыкальных данных 32
3.2 Алгоритм классификации 38
4 ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ 40
4.1 Последовательность и параметры обработки и анализа 40
4.2 Результаты 41
5 ЗАКЛЮЧЕНИЕ 43
6 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 44
Приложение А. Инструментарий и программная реализация 46

ПЕРЕЧЕНЬ ПРИНЯТЫХ ОБОЗНАЧЕНИЙ

HTM(Hierarchical Temporal Memory) – Иерархическая временная память.
MPF(Memory Prediction Framework) – Фреймворк Память-Предсказание.
MIR(Music Information Retrieval) – Получение Данных о Музыке.
KNN(K-Nearest Neighborhoods) - K-Ближайших Соседей.
DFT(Discrete Fourier transform) – Дискретное Преобразование Фурье
FFT(Fast Fourier Transform) – Быстрое Преобразование Фурье
СПМ – Спектральная Плотность Мощности

ВВЕДЕНИЕ

Очень много музыкальных данных создано и создается каждый день. Это все надо анализировать, сортировать, рекомендовать, рекламировать, продавать, искать, открывать новое, организовывать слушать, смешивать, создавать, генерировать [1] [2], автоматически и эффективно, на разных временных (цельная композиция или отдельная нота) и количественных (личные коллекции и многомиллионные) масштабах.

Сейчас это пытаются решить при помощи создания моделей каждой из составляющих музыки отдельно и получении данных непосредственно из сигнала [3] [4], либо используются социальные системы типа Last.fm[1] и Musicbrainz[2], основанные на метаданных, организациях частных коллекций и связей между людьми.

Во всех случаях используется очень много ручной работы и работы по переносу знаний экспертов в музыкальной области в область моделирования [5].

На момент 2010 года не существует систем, которые могли бы быть приемлемо универсальными, и масштабируемы в области музыки.

Возможным подходом для решения данной проблемы можно использовать алгоритмы, которые сами строят модели данных, которые им подаются.

HTM является таким алгоритмом, который успешно был применен в задач Компьютерного Зрения[3] [6], а теория в основе этого говорит о том, что алгоритмы могут быть применены успешно в задачах Компьютерного Слуха [7].

На данных момент опубликованы только попытки применения исследуемых алгоритмов к символьной музыкальной нотации [8] и построения специализированной версии алгоритмов для анализа музыки [9].

---

[1] http://last.fm – мультимедийная социальная сеть
[2] http://musicbrainz.org – хранилище музыкальных метаданных
[3] http://www.vitamind.com – приложение для слежения за объектами

ЦЕЛЬ И ЗАДАЧИ

Целью данной работы является создание системы для автоматического построения моделей музыкальных коллекций.

Задачи:

1) Получение оценки возможностей применения HTM в области MIR.
2) Повышение скорости и качества классификации музыкальных данных используя HTM.
3) Исследовать пути улучшения результатов путем синтеза мануальных моделей с построенными HTM и изменения HTM для качественного моделирования временных рядов.

1 ПРЕДМЕТНАЯ ОБЛАСТЬ - MIR

MIR – получение информации из музыки и о музыке автоматически, применение её для решения различных проблем.

На данных момент MIR активно развивается в научном, программно-техническом и коммерческом направлениях.

В данном разделе описаны решаемые MIR вопросы и использующиеся методы. Также приводятся основания того, почему HTM может быть применима в MIR.

1.1 Решаемые вопросы

Исследователи MIR пытаются решить ряд вопросов. Далее приведено перечисление некоторых из них:

1) Есть песня, нужно найти похожие.
2) Есть покупатель музыки, нужно порекомендовать ему ту которую он купит.
3) Есть производители музыки, им нужны мощные и эффективные средства для создания новой музыки.
4) Есть несколько друзей в социально сети слушающих музыку, нужно качественно порекомендовать что-то новое, чтобы они остались активными пользователями.
5) Есть слушатели музыки, которым недостаточно просто отображение музыки в виде списка [10] [1].
6) Есть работники индустрии развлечений, которым необходимо подбирать музыку, подходящую к конкретному моменту.
7) Есть исследователи музыки, которым необходим инструменты для проведения исследований.

1.2 Связь с другими областями науки

Основы MIR находятся в Компьютерный слух(Computer Audition). Далее приведен список смежных областей и решаемых ими вопросов:

1) Машинное обучение (Machine learning), Компьютерная Наука (Computer Science) , Компьютерный слух (Computer Audition) – создание математических и алгоритмических моделей музыки и понимания музыки человеком, автоматизация работы с музыкой.

2) Звуковой Инжиниринг (Audio Engineering), Обработка Сигналов(Signal processing) – обработка и предобработка звуковых данных.

3) Взаимодействие Человек-Компьютер(Human Computer Interaction), Компьютерное зрение(Computer Vision) , Визуализация (Visualization) – пытаются решить вопрос взаимодействия с музыкальными коллекциями и эффективного их изучения.

4) Получение Информации (Information Retrieval), Библиотечное дело (Library Science), Информатика (Information Science) – решают вопрос с организацией и поиском информации в больших коллекциях данных.
5) Музыкальное сознание (Music cognition), Психология Музыки (Music psychology), Теория Музыки (Music theory), Музыковедение (Musicology), Восприятие(Perception) – отвечают на вопросы “Что такое музыка?”, “Как музыка влияет на человека?”

[4] [5]

1.3 Способы описания музыки

Музыку можно описывать и оценивать с разных сторон:

1) Восприятия: энергия, текстура, ритм, темп, оркестровка, гармоничность, тон, характер мелодии.
2) Редакторская: импровизации, руководство по воспроизведению, оркестровка.
3) Библиографическая: название, исполнитель, композитор, страна, обложка альбома.
4) Культурная: жанр, субъективные квалификаторы.
5) Когнитивная: опыт, эмоции, ассоциации.
6) Символическая: мелодия, гармония, структура.
7) Текстовая: слова песни, информация об исполнителях [5].

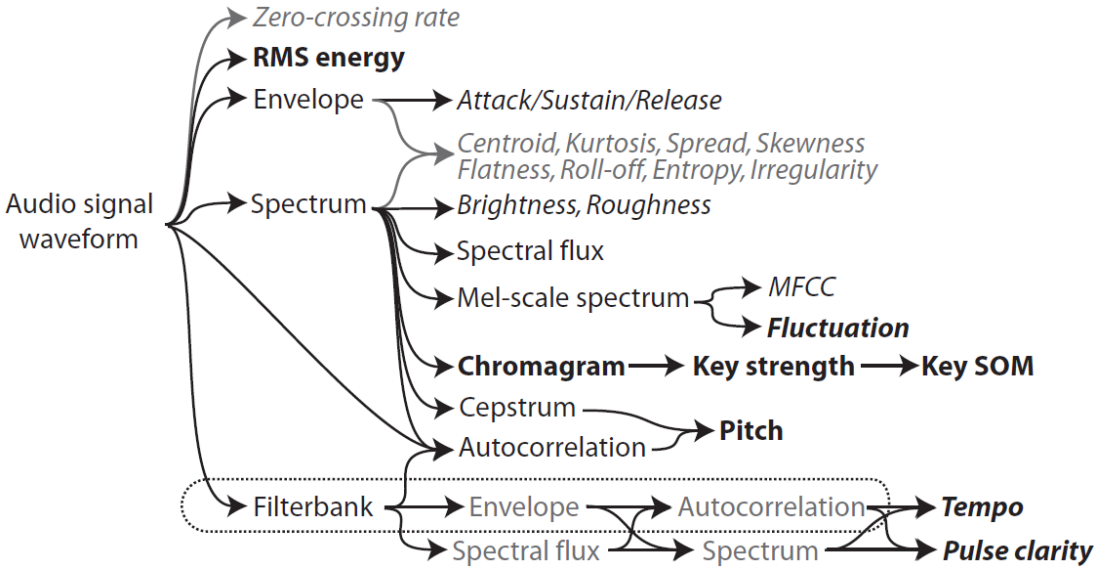
После специальной обработки эта информация может быть использована для решения различных задач.

1.4 Методы получения данных о музыке

Получить численные данные о музыке можно обработкой текстовых данные из интернета (контекст) либо из самого звука (содержимое). В данной работе использовались и описаны только содержательные методы.

Каждое музыкальное свойств относится к одному из измерений традиционно определенных в музыкальной теории. На Рис. 1

изображены возможные способы вычисления значений моделей различных низко- и среднеуровневых музыкальных структур.



“Boldface” *текст подсвечивает свойства относящиеся к тону и тональности(хромограмма, сила ноты, нотная SOM) и динамике(Root Mean Square(RMS), energy). “**Bold italics”** обозначают свойства относящиеся к ритму (темпу), четкости и отклонению пульсации. “Simple italics” подсвечивают большой набор свойств которые можно ассоциировать с тембром. “Grey italics” операции могут быть применены к различным представлениям: например, статистические моменты (centroid, kurtosis, etc.) могут быть применены как к спектру или огибающим, так и к гистограммам основанными на любом из данных свойств.*

Рис. 1 Содержательные методы MIR

Данные вычисленные по вышеприведенным моделям используются совместно с Машинным Обучения для определения того, что такое музыкальный стиль, жанр, настроение и т.д. Полученные результаты используются для аннотирования, тагифицирования, классификации и поиска.

1.5 Иерархичность музыкальных структур

Для анализа музыкальных композиций их разбивают на отрезки. Далее приведен возможный способ временного разбиения сигнала:

1) Окно анализа - характеристики сигнала примерно одинаковы (~30мс)
2) Текстурное окно – заметна текстура сигнала (~1000 мс)
3) Секция – сигнал можно назвать музыкой (~10с)
4) Композиция – весь сигнал.

[4]

В работе [2] обращено внимание на иерархию в возможно разбиении и вычисляемых музыкальных свойств. На Рис. 2 Декомпозиция музыки в древовидную структуру и Рис. 3 изображены возможные иерархии музыкальных свойств в области вычисления тембра.

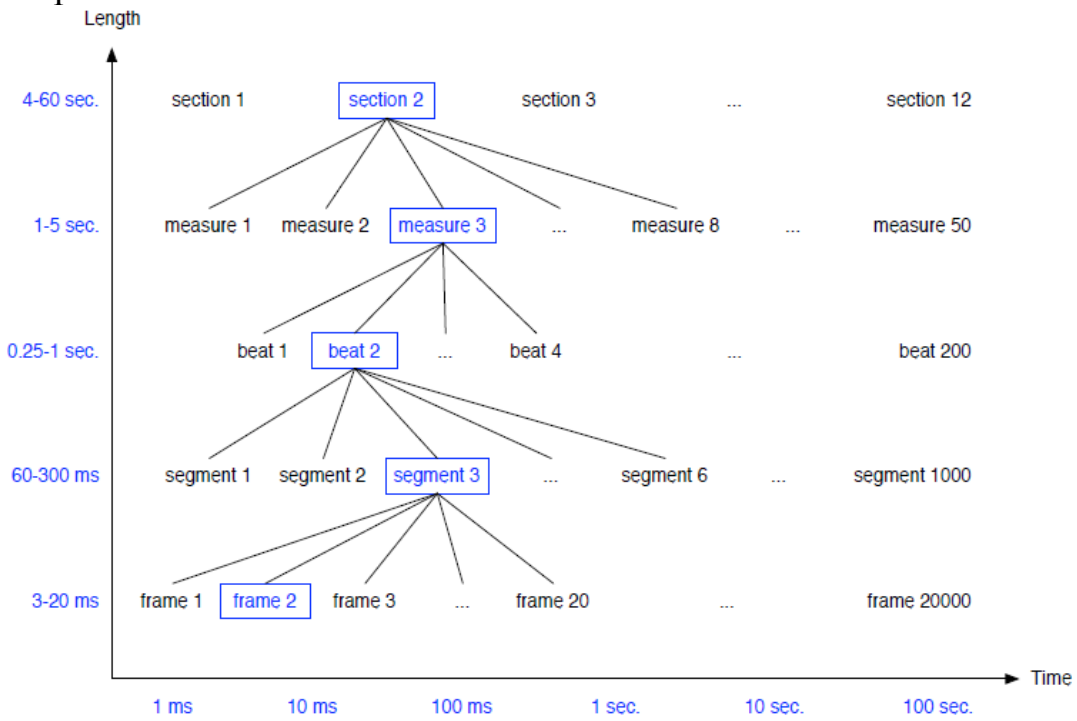
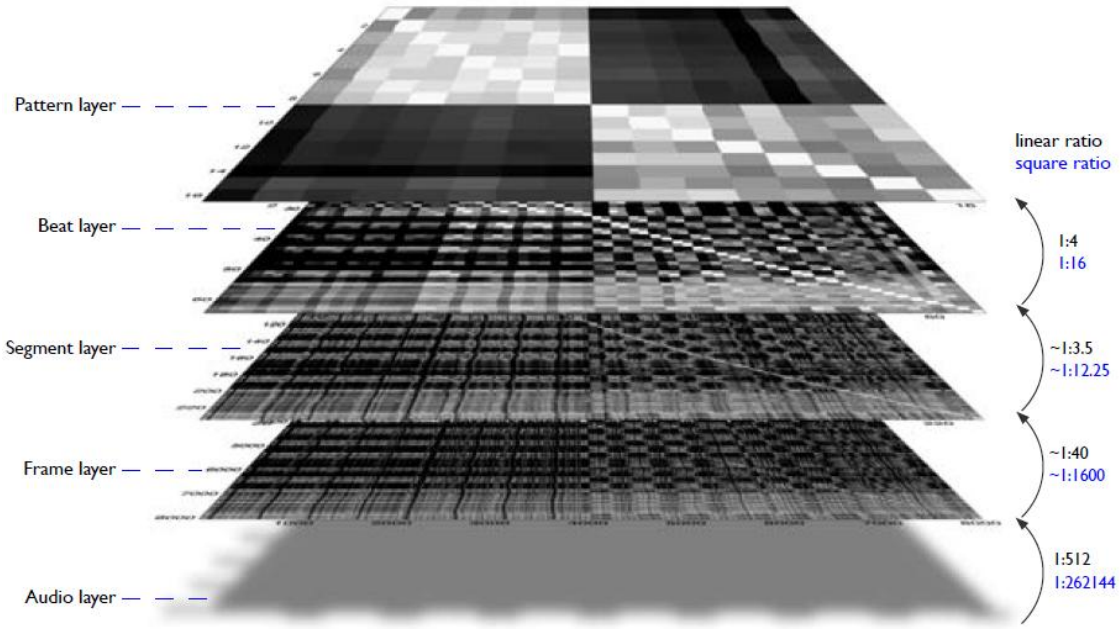


Рис. 2 Декомпозиция музыки в древовидную структуру

Структурные иерархии, найденные в частотной области (отношения между нотами, аккордами или тональностями) и во временной области (такт, ритмическая компоновка, паттерны, макроструктуры), открывают сложность и взаимосвязь между различными компонентами, составляющими музыку, и могут быть использованы для анализа.



Для каждого уровня временного разделения можно посчитать значение музыкального свойства основанного на понимании тембра, и найти матрицу самопохожести, которая часто используется в MIR. Такая иерархия позволяет уменьшить время и необходимую память для анализа, а также смотреть на анализ не с алгоритмической стороны, а с структурной.

Рис. 3 Использование иерархического представления для анализа

2 ИЕРАРХИЧЕСКАЯ ВРЕМЕННАЯ ПАМЯТЬ

HTM берет свои основы в MPF. MPF - теория о происхождении интеллекта и работе неокортекса, основными идеями которой являются [11] [7]:

1) Неокортекс конструирует модель пространственных и временных паттернов, которые ему предоставляются. Целью конструирования модели является предсказание следующего входящего паттерна.
2) Неокортекс сконструирован из повторяющихся вычислительных блоков известных как канонические кортикальные цепи. С вычислительной точки зрения, такая цепь может быть интерпретирована как узел повторенный несколько раз.
3) Неокортекс организован иерархически. Это значит что узлы (базовые вычислительные блоки) соединены в иерархию в форме дерева.
4) Функция неокортекса – моделировать мир, который ему показывается. Эта модель строиться, используя пространственную и временную иерархию путем запоминания паттернов и очередей в каждом из узлов иерархии. Эта модель используется для предсказаний о входных данных.
5) Неокортекс строит эту модель мира без учителя.
6) Каждый узел в иерархии хранит большое количество паттернов и очередей. Метод распознавания паттернов, используемый неокортексом, в значительной степени основан на хранении большого количества паттернов.
7) Выход каждого узла выражается в терминах очередей паттернов, которые он изучил.
8) Информация передается вверх и вниз по иерархии для распознавания и разрешения неоднозначности.

Существуют различные подходы и преследуемые цели в алгоритмической реализации MFP [12] [9]. HTM - попытка реализации MPF.

Подобные алгоритмы разделяют одно общее свойство – они симулируют сети подобные тем, что есть в кортексе [13].

2.1 Предисловие

Есть множество вещей, легко дающихся людям, но недоступных пока для компьютеров. Такие задачи, как распознавание визуальных

паттернов, понимание естественного языка, распознавание и манипулирование объектами на ощупь и ориентирование в сложном мире, элементарны для людей. Да, несмотря на десятилетия исследований, нет жизнеспособного алгоритма для реализации на компьютере этих и многих других когнитивных функций.

У человека за эти способности в основном отвечает неокортекс. НТМ – это технология, имитирующая структурные и алгоритмические свойства неокортекса.

Для традиционных компьютеров программисты пишут программы, решающие конкретные проблемы. Например, одна программа может быть использована для распознавания речи, а другая, совершенно отличная от нее программа, может быть использована для моделирования погоды. С другой стороны, об НТМ лучше думать, как о системе памяти. НТМ не программируются и не выполняют различные алгоритмы для различных проблем. Вместо этого, НТМ «учатся» решать проблему. НТМ обучают путем подачи на них сенсорных данных, и способности НТМ определяются в основном тем, какие данные подавались.

НТМ организованы как древовидная иерархия узлов, где каждый узел реализует общие функции обучения и памяти. НТМ хранит информацию в иерархии, моделируя мир. Все объекты в мире, будь то машины, люди, здания, речь или поток информации в компьютерной сети, имеют какую-либо структуру. Предполагается что эта структура иерархическая и в пространственном, и во временном отношении. НТМ также иерархическая и в пространстве и во времени, и, следовательно, может эффективно фиксировать и моделировать структуру мира.

НТМ похожи на Байесовские Сети; однако, они отличаются от большинства Байесовских Сетей тем, как используется время, иерархия и внимание.

Существуют две наиболее важные способности НТМ - способность обнаруживать причины и выдвигать гипотезы о причинах. Также НТМ может 2 другие особенности - предсказания и поведения.

2.2 Что делает НТМ

Уже свыше 25 лет известно, что неокортекс работает по единому алгоритму; зрение, слух, осязание, язык, поведение и многое другое, выполняемое неокортексом – проявления единого алгоритма,

применяемого к различным модальностям сенсорной информации. Это же верно и для НТМ. Так что, при описании того, что делает НТМ и как она работает, объяснение будет в терминах, не зависящих от сенсорной модальности.

НТМ выполняет следующие три основных функции в независимости от конкретной задачи, к которой она применяется.

1) Обнаружение причин в мире
2) Выдвижение гипотез о причинах новой информации
3) Предсказание

2.2.1 Обнаружение причин в мире

Рис. 4 Цель НТМ показывает, как НТМ взаимодействует с внешним миром. Слева на этом рисунке прямоугольник, представляющий мир, который изучает НТМ. Мир состоит из объектов и их отношений. Некоторые из объектов мира являются физическими, такие как автомобили, люди и дома. Некоторые из объектов мира могут быть не физическими, такие как мысли, слова, песни или потоки информации в сетях. Важным атрибутом объектов мира с точки зрения НТМ является то, что у них устойчивая структура; они существуют во времени. Назовем объекты мира «причинами». Можно понять, откуда взялось это слово, если зададитесь вопросом «какова была изначальная *причина* паттерна на сетчатке» или «какова была изначальная *причина* звука, услышанного ушами». В любой момент времени в мире активна иерархия причин. При восприятии устной речи, причинами звуков, поступающих в уши, являются фонемы, слова, фразы и идеи. Все они одновременно активны, и все они являются правомерными причинами слуховой информации.

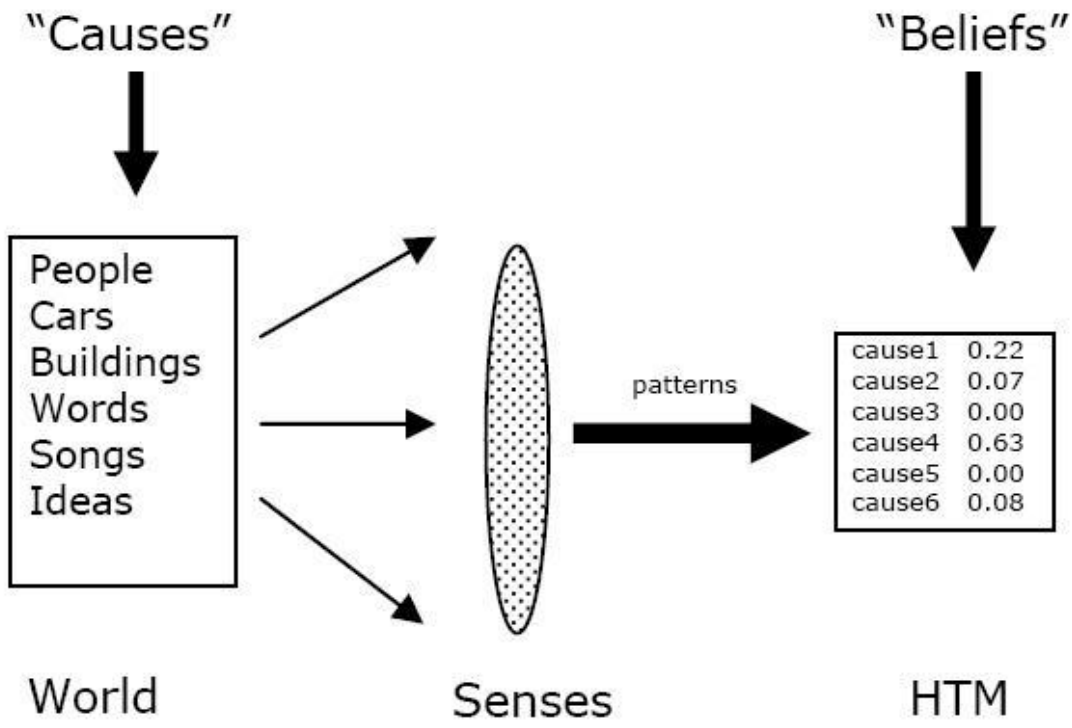


Рис. 4 Цель HTM

Определенные HTM могут быть нацелены только на подмножество мира. HTM может быть ограничена знанием о финансовом рынке, или взаимодействовать только с погодным феноменом, или только с геофизическими данными, демографическими данными или данными, собранными с сенсоров, установленных на автомобилях. Далее, при ссылке на «мир» HTM, будет имеется в виду его ограниченную часть, которая действует на HTM.

На правой стороне Рис. 4 изображена HTM. Она взаимодействует с ее миром через один или несколько сенсоров в средней части рисунка. Сенсоры делают выборки некоторых атрибутов мира, таких как свет или прикосновение, однако сенсоры, используемые HTM, не обязаны быть аналогичными органам чувств человека. Обычно сенсоры не обнаруживают объекты мира напрямую. Цель HTM - обнаружить в потоке сырой информации от сенсоров то, что существуют такие объекты, как «машина» или «слово». Сенсоры обычно подают в HTM массив данных, где каждый элемент – это измерение некоторого маленького атрибута мира. У человека зрительный нерв, переносящий информацию от сетчатки в кортекс, состоит приблизительно из миллиона волокон, где каждое волокно переносит информацию об освещенности маленькой части видимого пространства. В слуховом нерве около тридцати тысяч волокон, где

каждое волокно несет информацию о небольшом частотном диапазоне звукового спектра. Сенсоры, подключенные к НТМ, как правило, будут организованы аналогично. То есть, информация с сенсоров будет типологически выстроенной коллекцией входных данных, где каждый элемент измеряет локальную и простую величину.

У всех систем НТМ есть какой-либо тип сенсорной информации, даже если данные поступают из файла. С точки зрения НТМ, у сенсорных данных есть два основных свойства. Первое - сенсорные данные должны измерять что-то, что прямо или косвенно связано с причинами в мире, которые могли бы заинтересовать. Если необходимо, чтобы НТМ изучала погоду, она должна чувствовать что-то, относящееся к погоде, такое как температура и давление в различных местах. Если НТМ предназначена для анализа трафика в компьютерных сетях, она могла бы измерять количество пакетов в секунду и загрузку процессоров на маршрутизаторах. Второе – сенсорные данные должны поступать во времени непрерывным потоком, в то время как причины, лежащие в основе сенсорных данных, остаются относительно стабильными. Временной аспект сенсорных данных может исходить из движения или изменения объектов в реальном мире (такого, как движение автомобиля или ежеминутные флуктуации рынка ценных бумаг), или он может исходить из движения самой сенсорной системы в мире. В любом случае, сенсорные данные должны непрерывно изменяться во времени для того, чтоб НТМ обучалась.

НТМ получает пространственно-временные паттерны, приходящие от сенсоров. Поначалу у НТМ нет знания о причинах в мире, но через процесс обучения, который будет описан далее, она «открывает», что является причиной. Конечной целью этого процесса является то, что в НТМ образуется внутреннее представление причин в мире. В мозгу нервные клетки обучаются представлению причин в мире, например, нейроны, активизирующиеся, когда при виде лица. В НТМ причины представляются векторами чисел. В любой момент времени, основываясь на текущей и прошлых выборках, НТМ будет назначать вероятность, что в данный момент воспринимается та или иная причина. На выходе НТМ выдается набор вероятностей для каждой из изученных причин. Это распределение вероятностей в каждый момент называется «гипотезой». Если НТМ знает о десяти причинах в мире, у нее будет десять переменных, представляющих эти причины. Значения эти переменных – это гипотезы – то, что по мнению НТМ происходит в мире в данный момент. Обычно НТМ

знает о множестве причин и в действительности изучает иерархию причин.

Обнаружение причин это сердце восприятия, творчества и интеллекта. Ученые пытаются открыть причины физических феноменов. Бизнесмены ищут причины, лежащие в основе маркетинговых или бизнес-циклов. Врачи ищут причины болезней. С момента вашего рождения ваш мозг постепенно запоминает представления для любых вещей, которые вы, в конечном счете, познаёте. Вы должны обнаружить, что автомобили, здания, слова и мысли являются устойчивыми структурами мира. Прежде чем вы сможете узнавать что-то, ваш мозг должен сначала обнаружить, что существуют вещи.

Все системы НТМ должны пройти через фазу обучения, в которой НТМ изучает, какие причины существуют в мире. Сначала все НТМ изучают маленькие и простые причины их мира. Большие НТМ, когда предоставлено достаточное количество сенсорных данных, могут обнаружить сложные высокоуровневые причины. При достаточном обучении и правильном дизайне должно быть возможно построить НТМ, способные обнаружить причины, которые не могут обнаружить люди. После начального обучения НТМ может продолжить обучение или нет, в зависимости от прикладных нужд.

Просто обнаружение причин может иметь очень большое значение. Понимание высокоуровневых причин рыночных флуктуаций, болезней, погоды, доходов производства и отказов сложных систем, таких как энергетические сети, очень ценно. Обнаружение причин также необходимый предшественник для второй способности НТМ – выдвигать гипотезы.

2.2.2 Выдвижение гипотез о причинах новой информации

Когда НТМ знает, какие причины существуют в ее мире и как представлять их, она может выдвигать гипотезы. «Выдвижение гипотез» подобно распознаванию образов. При наличии нестандартной информации НТМ будет «выдвигать гипотезы» о том, какая из известных причин вероятнее всего присутствует в мире в данный момент. Например, если бы существовала система зрения, основанная на НТМ, то ей можно было бы показать ей изображения и она могла бы выдвинуть гипотезы о том, какие объекты на картинках. Результатом было бы распределение гипотез по всем известным причинам. Если бы

картинка была недвусмысленной, распределение гипотез было бы с ярко выраженным максимумом. Если бы картинка была сильно неоднозначна, распределение гипотез было бы плоским, поскольку НТМ не была бы уверенной, на что она смотрит.

Текущие гипотезы НТМ могут быть непосредственно считаны с системы, чтоб быть использованными где-то за пределами НТМ (что не возможно для человека). Или, текущие гипотезы могут быть использованы НТМ для того, чтоб делать предсказания или генерировать поведение.

В большинстве систем НТМ сенсорная информация всегда будет новой. В системе зрения, подсоединенной к камере, мог бы быть миллион пикселей сенсорной информации. Если камера будет смотреть на сцены из реального мира, маловероятно, что один и тот же паттерн попадет в НТМ дважды. Таким образом, НТМ должна манипулировать с новой информацией и при выдвижении гипотез, и во время обучения. Фактически, у НТМ нет отдельного режима, в котором бы она выдвигала гипотезы. НТМ всегда выдвигает гипотезы о причинах, даже в процессе обучения (хотя выводя плохо до того как пройдено достаточно длительное обучение). Как упоминалось ранее, существует возможность запретить способность к обучению по окончании процесса обучения с сохранением способности выдвигать гипотезы.

Большинству приложений НТМ будут требоваться изменяющиеся во времени сенсорные данные для того, чтоб выдвигать гипотезы, хотя некоторым – нет. Это зависит от природы сенсоров и причин. Эти различия можно увидеть у человека. Органы слуха и осязания без временной компоненты не могут выдвинуть гипотезу практически ни о чем. Со зрением ситуация двойственная. В отличие от ситуации с осязанием и слухом, люди могут распознавать изображения (то есть выдвигать гипотезы о причинах), когда изображение мелькает перед ними и глаза не успевают сдвинуться. Таким образом, визуальные гипотезы не всегда требуют изменяющейся во времени информации. Однако, в нормальном визуальном процессе глаза движутся, движется тело и объекты в мире также движутся. Таким образом, идентификация статических, мелькающих картинок – это специальный случай, возможный из-за статистических свойств зрения. В общем случае, даже в случае зрения, выдвижение гипотез происходит на изменяющейся во времени информации.

Хотя иногда возможно выдвижение гипотез на статических сенсорных паттернах, теория, лежащая в основе HTM показывает, что невозможно обнаружить причины, не имея непрерывно изменяющейся информации [11] [14] [7]. Таким образом, все системы HTM, даже те, которые выдвигают гипотезы на статических паттернах, должны обучаться на изменяющейся во времени информации. Недостаточно просто изменяющейся сенсорной информации, для этого было бы достаточно последовательности некоррелирующих паттернов. Обучение требует, чтобы в процессе поступления изменяющихся паттернов причина оставалась неизменной.

Выдвижение гипотез о новой информации является очень существенным. Есть множество задач распознавания паттернов, которые кажутся человеку простыми, но которые существующие компьютеры не могут решить. HTM может решать такие задачи быстро и точно. В дополнение есть множество задач по выдвижению гипотез, которые трудны для человека, но которые системы HTM могли бы решить.

2.2.3 Предсказания

HTM состоят из иерархии узлов памяти, где каждый узел изучает причины и формирует гипотезы. Часть алгоритма обучения, выполняемая каждым узлом, заключается в том, чтобы хранить возможные последовательности паттернов. Комбинируя память о вероятных последовательностях с поступающей информацией, каждый узел может совершать предсказания того, что вероятно должно произойти дальше. HTM в целом, как набор узлов, также делает предсказания. Точно также, как HTM может выдвигать гипотезы о причинах новой информации, она также может предсказывать новые события. Предсказание будущих событий – это суть творчества и планирования. Оставив на потом детали того, как это работает, сейчас можно сформулировать для чего может быть использовано предсказание. Есть несколько применений предсказания в HTM, включая установку предпочтений, воображение и планирование, а также генерацию поведения.

Установка предпочтений

Когда HTM предсказывает, что вероятнее всего должно произойти далее, предсказание может выступать в качестве того, что называется «априорная вероятность», обозначая, что оно склоняет систему выдвигать гипотезы по предсказанным причинам. Например, если

НТМ будет обрабатывать текст или устную речь, он мог бы автоматически предсказывать, какие звуки, слова и мысли скорее всего возникнут далее. Это предсказание помогает системе понимать зашумленные или недостающие данные. Если возникает звуковая неоднозначность, НТМ будет интерпретировать звук, основываясь на том, что ожидается.

В случае НТМ есть возможность вручную установить априорные вероятности в дополнение к априорным вероятностям, установленным через предсказание. То есть, можно вручную сказать НТМ, чтобы она ожидала или искала определенные причины или набор причин, таким образом реализуя направленный поиск.

2.3 Обнаружение причин и выдвижение гипотез НТМ

НТМ структурно устроены как иерархия узлов, где каждый узел выполняет один и тот же алгоритм. На Рис. 5 изображена простая иерархия НТМ. Сенсорные данные поступают снизу. Сверху выходит вектор, в котором каждый элемент представляет потенциальную причину сенсорных данных. Каждый узел иерархии выполняет ту же функцию, что и вся иерархия целиком. То есть, каждый узел рассматривает пространственно-временные паттерны, поступающие в него и обучается назначать причины поступающим в него паттернам. Проще говоря, каждый узел в независимости от его места в иерархии открывает причины своей входной информации.

Выход каждого узла одного уровня становится входом следующего уровня иерархии. Самые нижние узлы иерархии получают информацию от маленьких участков сенсорной области. Следовательно, причины, которые он открывает, относятся к маленькой части области сенсорных входов. Вышестоящие области получают информацию от многочисленных нижестоящих узлов, и снова открывают причины этой информации. Эти причины промежуточной сложности, возникающие на участках большего размера. Узел или узлы на верхушке иерархии представляют высокоуровневые причины, которые могут возникать в любой части сенсорного поля. Например, в НТМ, выдвигающей визуальные гипотезы, узлы внизу иерархии обычно обнаруживают простые причины, такие как края, линии и углы на маленькой части визуального пространства. Узлы на верхушке иерархии будут представлять сложные причины, такие как собаки, лица, автомобили, которые могут появиться на всем визуальном пространстве или на

любой части визуального пространства. Узлы промежуточных уровней иерархии представляют причины промежуточной сложности, которые возникают на участках промежуточного размера. Помните, что все эти причины должны быть обнаружены НТМ. Они не программируются и не выбираются разработчиком.

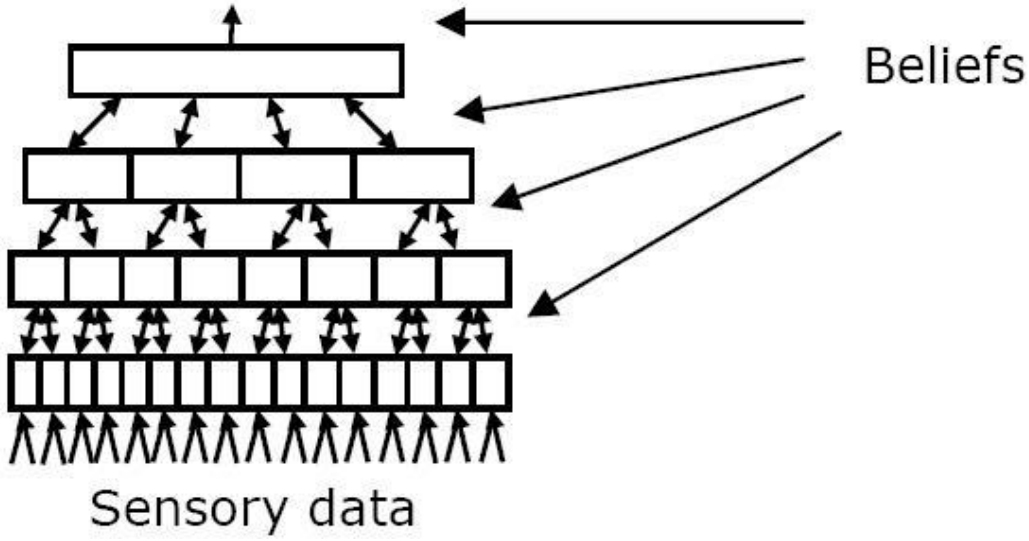


Рис. 5 Иерархия узлов НТМ

В НТМ гипотезы существуют на всех уровнях иерархии, не только на верхнем уровне. Гипотеза – это внутреннее состояние каждого узла. Её можно рассматривать как вектор чисел, каждое из которых представляет вероятность того, что причина активна.

Каждый элемент в векторе гипотезы (то есть каждая причина) зависит сам от себя. Каждая причина может быть понята и интерпретирована сама по себе и имеет свое собственное значение. Другими словами, значение переменной, представляющей причину, не зависит от того, какие другие причины могут быть активными в том же самом векторе. Это не значит, что причины, представляемые узлом, статистически независимы, или что только одна из них активна за раз. Несколько причин могут быть активными одновременно. Представления, используемые в НТМ, отличаются от тех, которые, скажем, используются в ASCII коде. Конкретный бит из восьми битов ASCII кода не имеет значения сам по себе.

Выход узла – также вектор. Выход аналогичен гипотезам узла, и наследуется от вектора гипотез. Сейчас выход узла будет рассматриваться как его гипотезы. Хотя это не совсем корректно, так будет проще описывать операции в НТМ.

Предполагая это, можно сказать, что информация на входе узла – это гипотезы дочерних узлов. Выход узла это гипотезы, которые станут частью информации, поступающей на вход родителя. Даже будет корректным рассматривать информацию от сенсоров как гипотезы, приходящие от сенсорной системы.

В идеальном мире не было бы никакой неопределенности на каждом узле. Однако, на практике такого не бывает. Одним из важных свойств HTM является то, что она быстро разрешает конфликт или неоднозначность входной информации по мере ее продвижения вверх по иерархии.

Каждый узел HTM имеет обычно фиксированное количество причин и фиксированное количество выходных переменных. Следовательно, HTM начинает с фиксированного количества возможных причин, и, в процессе тренировки она обучается присваивать им смысл. Узлы не «добавляют» причины по мере открытия, вместо этого, в процессе обучения, смысл выходных переменных постепенно изменяется. Это происходит на всех уровнях иерархии одновременно. Следствием такой методики обучения является то, что необученная HTM не может формировать осмысленных представлений на вершине иерархии до тех пор, пока узлы на нижнем уровне не пройдут достаточное обучение.

Базовые операции в каждом узле делятся на два шага. Первый шаг – связать входной паттерн узла с одной из множества точек квантования (представляющих обобщенные пространственные паттерны входной информации). Если у узла есть 100 точек квантования, узел назначает каждой из 100 точек квантования вероятность того, что текущая входная информация соответствует точке квантования. Снова, на этом первом шаге, узел определяет, насколько близко к каждой из точек квантования текущий входной паттерн, и назначает вероятность каждой точке квантования.

На втором шаге узел ищет обобщенные последовательности этих точек квантования. Узел представляет каждую последовательность переменной. По мере поступления паттернов во времени, узел назначает этим переменным вероятность, что текущая входная информация является частью каждой из последовательностей. Набор этих переменных для последовательностей является выходом узла и передается вверх по иерархии родительскому(им) узлу(ам).

Узел также может посылать информацию своим детям. Сообщение, идущее вниз по иерархии, представляет распределение по точкам квантования, тогда как сообщение, идущее вверх по иерархии, представляет распределение по последовательностям. Следовательно, по мере продвижения информации вверх по иерархии каждый узел пытается слить серию входных паттернов в относительно стабильный выходной паттерн. По мере продвижения информации вниз по иерархии каждый узел принимает относительно стабильные паттерны от своих родителей и пытается обратить в последовательности пространственных паттернов.

Путем сопоставления причин последовательностям паттернов происходит естественное слияние времени по мере продвижения паттерна вверх по иерархии. Быстро изменяющиеся низкоуровневые паттерны становятся медленно изменяющимися по мере их продвижения вверх. Обратное также верно. Относительно стабильные паттерны на верхушке иерархии может развернуться в сложную временную последовательность паттернов внизу иерархии. Изменяющиеся паттерны, поступающие на вход узла, аналогичны сериям музыкальных нот. Последовательности этих нот подобны мелодиям. Если входной поток, поступающий на узел, соответствует одной из запомненных им мелодий, узел передает «название» мелодии вверх по иерархии, а не отдельные ноты. Следующий вышестоящий узел делает то же самое, рассматривая последовательности последовательностей, и т.д. Каждый узел предсказывает, какая нота или ноты вероятнее всего должны последовать далее, и эти предсказания передаются вниз по иерархии дочерним узлам.

Количество уровней иерархии, количество узлов на каждом уровне иерархии и емкость каждого узла не критичны для основных положений теории НТМ. Аналогично, точные соединения между узлами не критичны, пока между каждыми двумя узлами сохраняются ясные отношения отцы/дети в иерархии. Рис. 6 показывает несколько вариаций соединений, которые правомерны в НТМ. Дизайн и емкость конкретных НТМ должна соответствовать поставленной задаче и доступным вычислительным ресурсам. Большая часть усилий может уйти на получение оптимальной производительности. Однако любые конфигурации НТМ будут работать в какой-либо степени. В этом отношении система надежна.

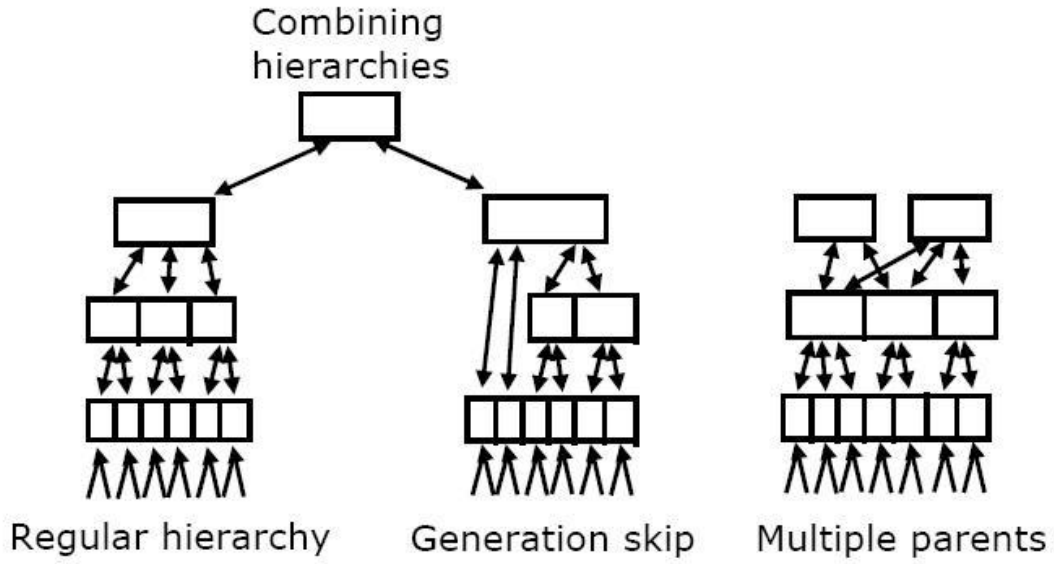


Рис. 6 Возможные иерархии

2.4 О важности иерархии

Есть несколько причин, почему так важно использовать иерархию узлов:

1) Совместное использование представлений изученных паттернов сокращает требования к количеству памяти и времени на обучение.
2) Иерархическая структура мира (в пространстве и времени) отражается иерархической структурой НТМ.
3) Методики Распространения Гипотез гарантируют быстрое достижение взаимно совместимых наборов гипотез в сети.

Теоретические и философские идеи лежащие в основе данных предположений описаны в [7]. Также в описана модель Иерархического Временного Мира, который является отражением НТМ которое порождает паттерны, и модель которого НТМ должен выучить [11].

2.5 Обнаружение причин и выдвижение гипотез отдельными узлами

Узел НТМ не «знает», что он делает. Он не знает, представляет ли получаемая информация свет, сигналы сонара, экономические данные, слова или данные производственного процесса. Узел также не знает, где он располагается в иерархии.

«Причина» - это всего лишь постоянная или повторяющаяся структура в мире. Таким образом, узел пытается назначить причины повторяющимся паттернам в его входном потоке. Есть два основных типа паттернов, пространственные и временные. Предположим, у узла есть сотня входов, и два из этих входов, i1 и i2 стали активными одновременно. Если это происходит достаточно часто (гораздо чаще, чем просто случайно), то можно предположить, что у i1 и i2 общая причина. Это просто здравый смысл. Если вещи часто возникают вместе, можно предположить, что у них общая причина где-то в мире. Другие частые пространственные паттерны могли бы включать десятки причин, возникающих вместе. Скажем, узел идентифицирует пятьдесят частых пространственных паттернов, найденных во входной информации. Когда приходит новый и оригинальный паттерн, узел определяет, насколько близко новый паттерн к ранее изученным 50 паттернам. Узел назначает вероятности, что новый паттерн соответствует каждому из 50 известных паттернов. Эти пространственные паттерны являются точками квантования, которые обсуждались ранее.

Пусть изученные пространственные паттерны будут иметь имена от sp1 до sp50. Предположим, узел наблюдает, что с течением времени sp4 чаще всего следует за sp7, и это происходит гораздо чаще, чем может позволить случай. Затем узел может предположить, что временной паттерн sp7-sp4 имеет общую причину. Это также здравый смысл. Если паттерны постоянно следуют один за другим во времени, то они вероятнее всего разделяют общую причину где-то в мире. Предположим, узел хранит 100 обобщенных временных последовательностей. Вероятность того, что каждая из этих последовательностей активна – это выходная информация этого узла. Эти 100 последовательностей представляют 100 причин, изученных этим узлом.

В каждый момент времени узел в HTM смотрит на входную информацию и назначает вероятность того, что эта входная информация соответствует каждому элементу из множества чаще всего возникающих пространственных паттернов. Затем узел берет это распределение вероятностей и комбинирует его с предыдущим состоянием, чтобы назначить вероятность того, что текущая входящая информация является частью множества чаще всего возникающих временных последовательностей. Распределение по множеству последовательностей является выходной информацией узла и передается вверх по иерархии. В конечном счете, если узел продолжает обучаться, то он мог бы модифицировать множество сохраненных

пространственных и временных паттернов для того, чтоб наиболее точно отражать новую информацию.

Для иллюстрации можно привести пример из зрительной системы. Рис. 7 показывает входные паттерны, которые узел мог бы ранее видеть на первом уровне гипотетической визуальной HTM. Этот узел имеет 16 входов, представляющих клочок двоичной картинки 4x4 пикселя. Рисунок показывает несколько возможных паттернов, которые могли бы появиться на этом клочке пикселей. Некоторые их этих паттернов более вероятны, чем другие. Можно предположить, что паттерны, которые могли бы быть частью линии или угла будут более вероятными, чем паттерны, выглядящие случайно. Однако, узел не знает, что он смотрин на клочок 4x4 пикселя и не имеет встроенных знаний о том, какие паттерны встречаются чаще и что они могли бы «обозначать». Все, что он видит – это 16 входов, которые могут принимать значения между 0 и 1. Он смотрит на свои входы в течение некоторого промежутка времени и пытается определить, каике паттерны наиболее часто встречаются. Он хранит представления для частых паттернов. Разработчик HTM назначает количество пространственных паттернов, или точек квантования, которые данный узел может представлять.

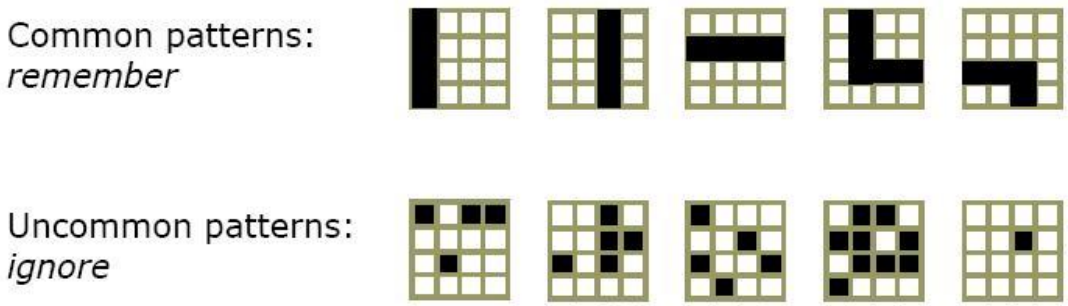


Рис. 7 Память точек квантования

Рис. 8 показывает три последовательности пространственных паттернов, которые могли бы видеть низкоуровневые визуальные узлы. Первые две строки паттернов являются последовательностями, которые будут вероятнее всего чаще встречаемыми. Можно видеть, что они представляют линию, движущуюся слева направо и угол, двигающийся из левого верхнего угла в правый нижний. Третья строка это последовательность паттернов, которая маловероятно попадалась узлу. И снова, у узла нет способа знать, ни то, какие последовательности наиболее вероятны, ни то, что они могли бы обозначать. Все, что он может сделать – это попытаться запомнить наиболее часто встречающиеся последовательности.

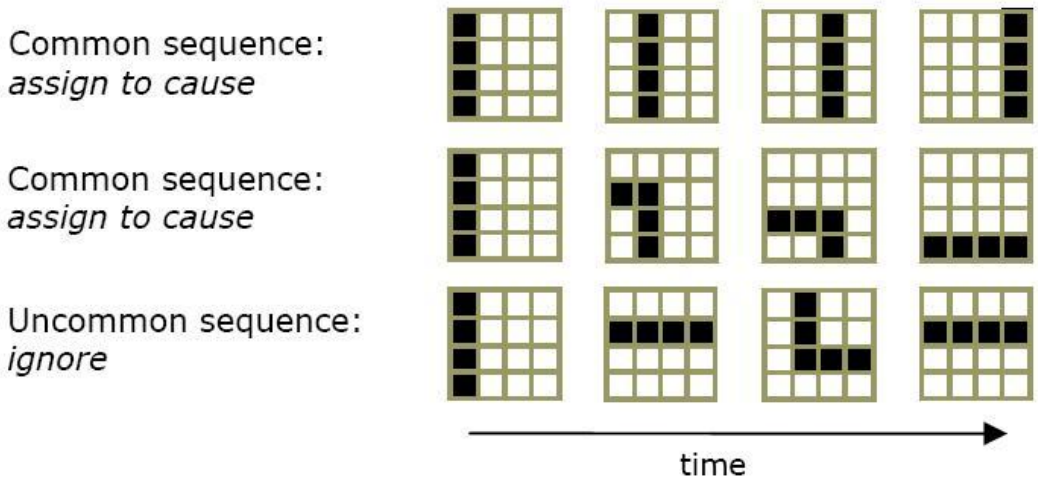


Рис. 8 Память последовательностей

В вышеприведенном примере не проиллюстрирована другая общая (но не всегда необходимая) функция узла. Если HTM делает предсказания, она использует свою память последовательностей для предсказания того, какие пространственные паттерны вероятнее всего появятся далее. Это предсказание в форме распределения вероятностей по изученным пространственным паттернам передается вниз по иерархии к дочерним узлам. Предсказание выступает в качестве «предпочтения», влияющего на нижестоящий узел.

Таким образом, можно сказать, что каждый узел в HTM сначала учится представлять наиболее часто встречающиеся пространственные паттерны во входном потоке. Затем он обучается представлять наиболее часто возникающие последовательности таких пространственных паттернов. Выход узла, идущий вверх по иерархии, является набором переменных, представляющих последовательности, или, более точно, вероятности того, что эти последовательности активны в данный момент времени. Узел также может передавать предсказываемые пространственные паттерны вниз по иерархии.

Обработка распределений и данных из реального мира

Паттерны на предыдущих рисунках были не реалистичными. Большинство будет иметь более 16 входных линий, и, следовательно, входные паттерны, получаемые узлом, смотрящим в реальный мир будут гораздо больше и практически никогда не будут повторяться. К тому же, элементы входных данных в общем случае имеют непрерывную градацию. Следовательно, узел должен иметь способность решить, какими являются обобщенные пространственные

паттерны, даже если ни один из паттернов не повторялся дважды и ни один из паттернов не был «чистым».

Похожая проблема существует и в отношении временных паттернов. Узел должен определить обобщенные последовательности пространственных паттернов, но должен делать это, основываясь на распределении пространственных паттернов.

Тот факт, что узел всегда видит распределения, обозначает, что в общем случае надо не просто перечислить и сосчитать пространственные и временные паттерны. Должны быть использованы вероятностные методики. Например, идея последовательности в HTM в общем случае не такая ясная, как последовательность нот в мелодии. В мелодии вы можете точно сказать, какой длины последовательность и сколько в ней элементов (нот). Но для большинства причин в мире не ясно, когда начинается и заканчивается последовательность, и существуют возможные разветвления на любом элементе. Аналогией могло бы быть хождение по улицам знакомого города. Выбираемый вами путь – это последовательность событий. Однако, нет набора путей по городу. «Последовательность» улиц в городе может изменяться, поскольку вы можете повернуть направо или налево на любом перекрестке. Также, не очевидны начало и конец последовательности. Но вы всегда знаете, в каком месте города вы находитесь. Пока вы передвигаетесь по улицам, вы уверены, что находитесь в том же самом городе.

Другая проблема возникает сама по себе при изучении последовательностей. Для того, чтобы сформировать последовательность, узел должен знать, когда приходит новый паттерн, то есть, где начинается элемент последовательности. Например, когда вы слышите мелодию, каждая новая нота имеет резкое начало, которое четко отмечает начало нового пространственного паттерна. Мелодия – это последовательность пространственных паттернов, где каждый паттерн отмечен началом новой ноты. Некоторые сенсорные паттерны подобны мелодиям, но большинство – нет. Если вы медленно вращаете объект, когда смотрите на него, сложно сказать, когда возникает новый пространственный паттерн. Узлы HTM должны решать, когда изменение во входном паттерне достаточно для того, чтоб отметить начало нового элемента.

Существует много прототипов того, как изучать пространственные паттерны с искаженными данными из реального

мира. Некоторые из этих моделей пытаются смоделировать часть визуального кортекса [15]. Гораздо меньше прототипов того, как изучать последовательности распределений, по крайней мере, не таким образом, как это работает в HTM.

2.6 О необходимости времени для обучения

HTM могла бы выдвигать гипотезы относительно «статических» сенсорных паттернов; основным примером может являться зрение. Однако, для обучения необходима изменяющаяся во времени информация. Даже системе статического зрения необходимо подавать изображения объектов, движущиеся в визуальном поле, чтоб она правильно обучалась, чтобы обнаруживала причины.

Обучение требует изменяющейся во времени информации поскольку каждый узел запоминает обобщенные последовательности паттернов, единственный способ сделать это – предоставить узлу последовательности паттернов во времени.

На самом базовом уровне, распознавание паттернов влечет за собой отнесение неизвестного входного паттерна к одной из нескольких категорий. Пусть, у нас есть система искусственного зрения, умеющая распознавать 1000 объектов или категорий. Существует практически неограниченное число возможных изображений, которые можно показать нашей системе и надеяться, что она отнесет каждое неизвестное изображение к правильной категории. Если «лошадь» - это одна из категорий, которые может распознать наша система, есть миллиарды визуальных паттернов, в которых вы немедленно бы распознали «лошадь». Мы хотели бы, чтобы наша система искусственного зрения делала то же самое.

Следовательно, распознавание паттернов это задача отображения «многие-к-одному». В одну категорию отображаются множество входных паттернов. Введем новый термин для описания отображения многие-к-одному: «пулинг»(“pooling”). Пулинг обозначает назначение одного обозначения многим паттернам, то есть складывание их в один и тот же пул(pool).

Каждый узел в HTM должен выполнять пулинг, поскольку иерархия в целом предназначена для выдвижения гипотез. Каждый узел должен делать это, даже если он просто распознает пространственные паттерны. Уже было показано два механизма для пулинга, хотя они так и не назывались. Пространственное квантование – это механизм пулинга, основанный на пространственном подобии. В

этом случае берется неизвестный паттерн и определяется, насколько близко он к каждой из точек квантования. Два паттерна, которые в достаточной степени «перекрываются», рассматриваются как один и тот же. Таким образом, множество возможных паттернов объединяются в каждую точку квантования. Это слабая форма пулинга, и сама по себе она недостаточна для решения большинства задач по выдвижению гипотез. Второй метод пулинга – это изучение последовательностей. В этом случае узел отображает множество точек квантования в единую последовательность. Этот метод пулинга более мощный, поскольку он допускает произвольное отображение. Он позволяет узлу группировать различные паттерны, которые пространственно не перекрываются. Он допускает произвольные отображения многие-к-одному.

Рассмотрим, например, распознавание изображения арбуза. Арбуз снаружи совсем не похож на внутренности арбуза. Нет существенного «пространственного» перекрытия между двумя этими изображениями или даже между частями изображений, так что в этом смысле это «произвольное» отображение. HTM использует время для пуллинга различных паттернов в одну группу. При вращении среза арбуза идет непрерывный поток паттернов, который изменятся от созерцания наружной стороны арбуза к созерцанию внутренностей. Самая исходная причина, «арбуз», остается постоянной во времени по мере того, как паттерны изменяются.

Конечно, ни один узел в HTM не помнит всю последовательность целиком. Узлы внизу иерархии запоминают довольно короткие последовательности, что вызвано небольшой областью паттернов, которые он может увидеть. Узлы на следующих уровнях более стабильны. Они запоминают последовательности последовательностей от предыдущего уровня. Стабильность возрастает по мере того, как паттерны поднимаются по иерархии. При достаточном обучении можно обнаружить, что выход самого высшего узла иерархии остается стабильным в течение всей последовательности. Каждый узел в иерархии выполняет пространственный и временной пулинг. Без временного пулинга HTM сама по себе не смогла бы узнать, что наружная сторона и внутренности арбуза имеют общую причину.

Этот аргумент верен для большинства возможных высокоуровневых причин в мире, являются ли они по своей сути временными (такие как речь, музыка и погода), или могут быть распознаны статически (такие, как зрение). Таким образом,

изменяющаяся во времени информация необходима для изучения причин мира.

Роль учителя

Не гарантируется, что HTM всегда будет изучать желаемые причины путем ощущения последовательностей входных паттернов. Например, что если HTM сначала была натренирована на внутренностях арбуза, а затем на его наружной стороне, то было бы естественным отнести эти отдельные паттерны к различным причинам. Следовательно, мы обучаем HTM на срезе арбуза, когда ей показывают последовательность от внутренностей до наружной стороны. Возможно, что HTM сформирует высокоуровневую причину (арбуз), которая представляет пулинг двух низкоуровневых причин (красные внутренности становятся зеленой наружной стороной). Но, также возможно, что этого не случится. Это зависит от дизайна иерархии, от того, как мы обучаем HTM и от статистики входной информации. Если переход между внутренностями и наружной стороной займет много времени, или если будет слишком много промежуточных шагов, может получиться, что HTM не совершит пулинг причин так, как ожидается.

Обучение корректной категоризации, то есть обучение корректным причинам может быть выполнено гораздо быстрее и определеннее при использовании обучения с учителем. В HTM это делается путем установки предварительных ожиданий на узлах верхнего уровня иерархии в процессе обучения.

Представление времени

Для некоторых временных паттернов важно конкретное или относительное время между элементами в последовательности. Например, временные интервалы между нотами в мелодии или время между фонемами в произносимом слове – важная часть этих причин. Биологический мозг имеет возможность изучать последовательности и с конкретной временной информацией, и без. Если в последовательности есть устойчивые временные интервалы, мозг запоминает их. Если устойчивых временных интервалов нет, мозг хранит последовательность без времени. В [7] было сделано предположение о том, как мозг хранит эту временную информацию. HTM нужен эквивалентный механизм для обнаружения и распознавания причин, которые требуют конкретной временной информации.

3 МЕТОДЫ ПРЕДОБРАБОТКИ И КЛАССИФИКАЦИИ ДАННЫХ

В данной главе описаны методы, используемые в данной работе для предобработки звука и классификации полученных векторов признаков.

3.1 Предобработка музыкальных данных

Звуковой моно сигнал конвертируется в 64ех полосную логарифмическую СПМ взятую с частотой 100 окон на секунду записи. Фильтры СПМ расположены логарифмически линейно и являются простой имитацией поведения внутреннего уха человека. Шагами трансформации являются:

1. Чтение аудио сигнал из файла. Аудио сигналы изначально читаются как Pulse Code Modulation (PCM) данные в форму очень длинных временных последовательностей (Рис. 9). Потом сигнал суммируется в один сигнал, если аудио в стерео формате.

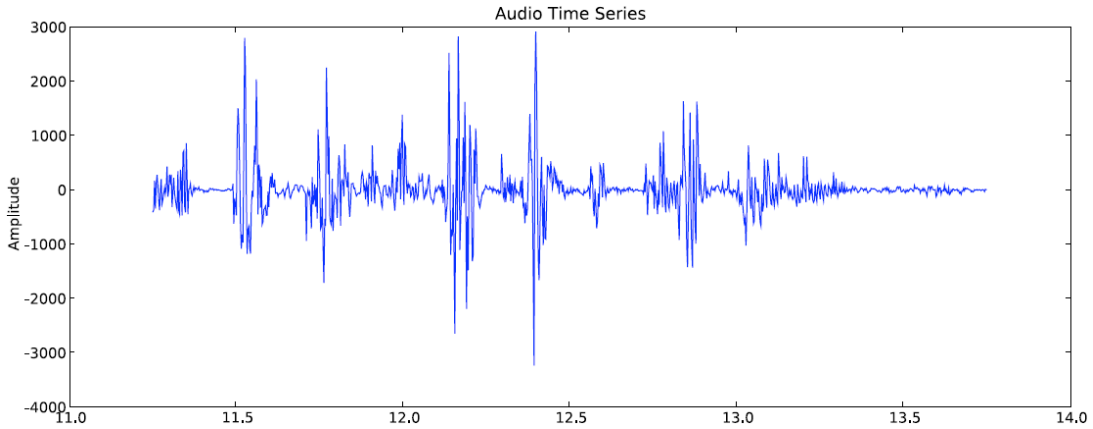


Рис. 9 Сигнал

2. Если необходимо, сигнал прореживается до 8 KHz.

3. Сигнал разделяется в перекрывающиеся окна размером в 512 значений. Первое окно формируется из первых 512 значений аудио данных. Последующее окно формируется отбрасыванием самых старых 80 значений. На Рис. 10, первое окно показано временными отсчетами соединенными серыми линиями, а второе окно отрисовано поверх черными линиями. Последующие окна сдвинуты на 80 отсчетов. При 8 KHz частоте выборки, окно размером 512 значений представляется 0.064 секунды, новые окна генерируется с частотой 100 на секунду сигнала (благодаря сдвигу на 80 отсчетов).

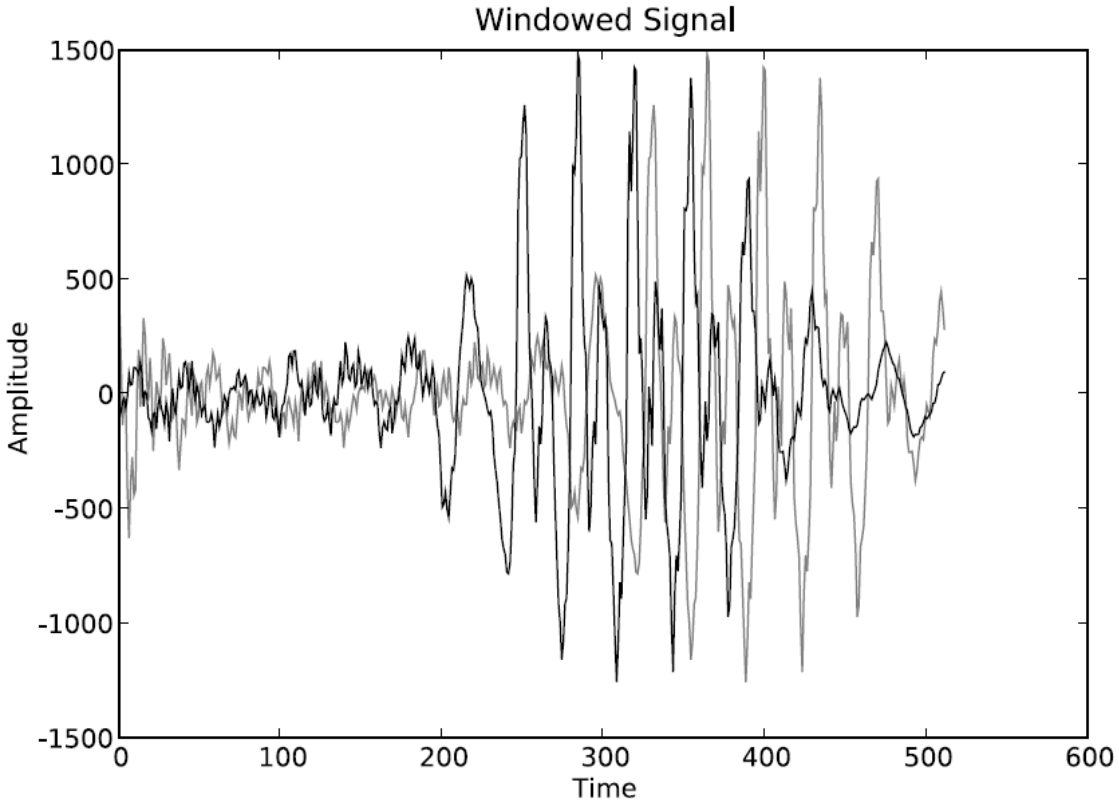


Рис. 10 Соседние блоки сигнала

4. Каждое окно трансформируется в СПМ используя кратковременное оконное DFT. Для соответствия предположениям DFT, каждое окно фильтруется так, что оно будет казаться периодическим для трансформации. 512 точечное окно Хэмминга используется, чтобы установить значения начала и конца окна в 0. (Показано на Рис. 11).

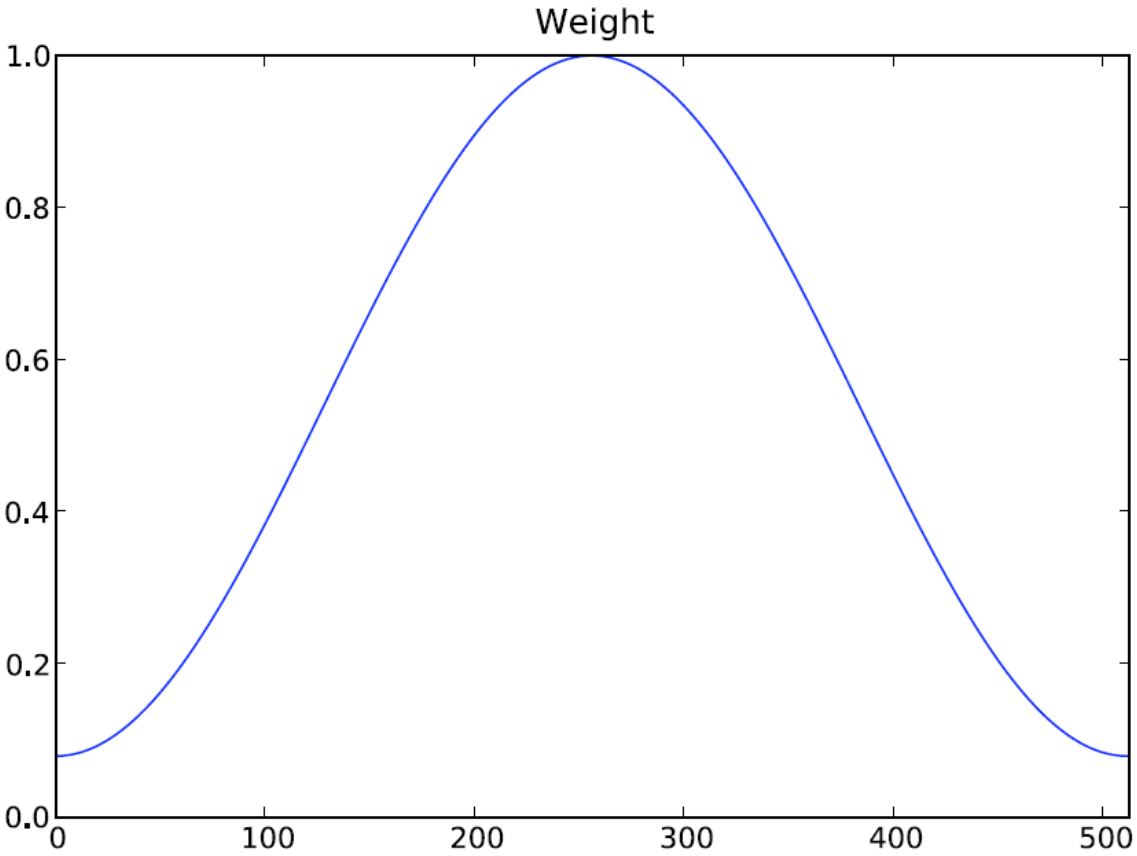


Рис. 11 Оконная функция Хэмминга

После применения окна Хэмминга, самая мощная часть сигнала для трансформации будет находиться во времени соответствующем в центру окна.

DFT вычисляется используя веществennозначное FFT, которое эффективно оперирует над блоками размера 512. Результат веществennозначного FFT есть спектр, содержащий вещественные DC значения (из-за наличия постоянной составляющей) и 256 комплексных значений, где каждое значение соответствует частотному карману. Комплексные значения могут быть преобразованы в амплитуду и фазу соответствующего частотного кармана. Использованная конфигурация алгоритмов оставляет только амплитуду, возводя её в квадрат для получения СПМ. Результат, в логарифмическом масштабе, можно видеть на Рис. 12 (более светлые линии представляют собой FFT предыдущих окон, более темная – текущего окна). Множество последовательно взятых спектров организованных во времени и представленных как изображение являются спектрограммой (Рис. 13)

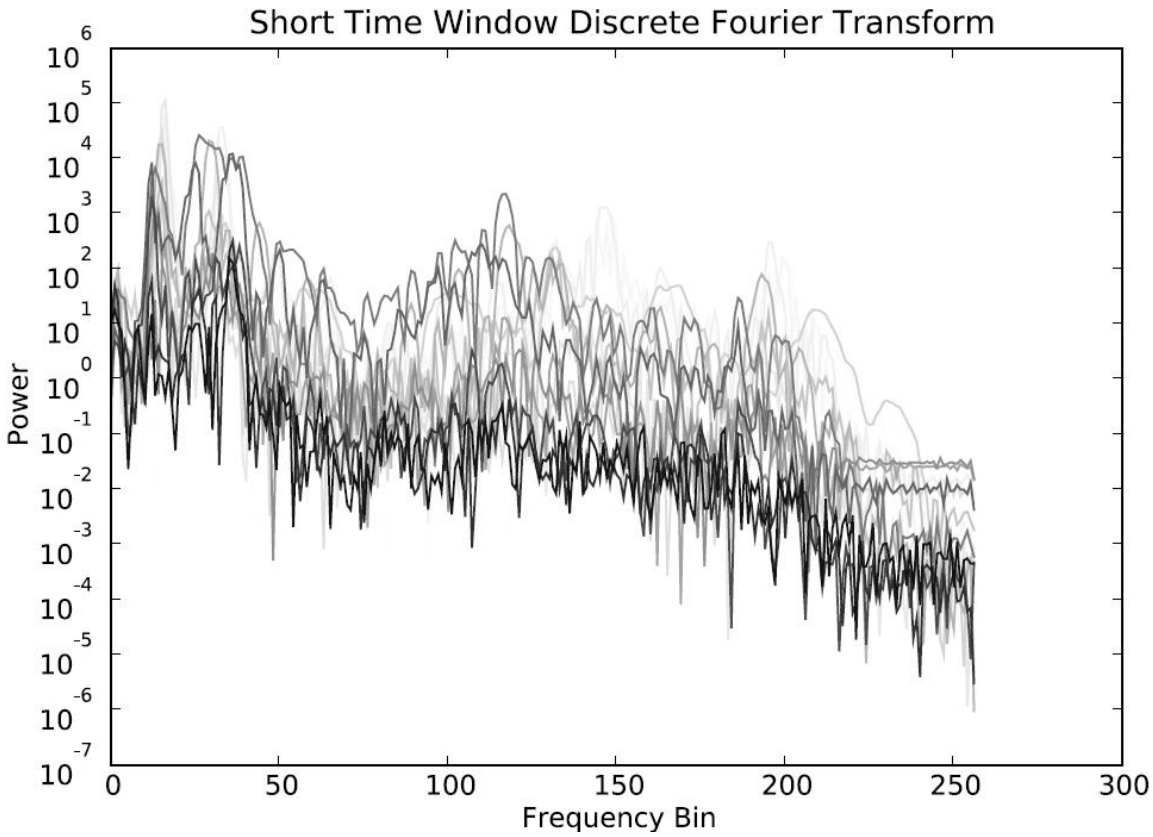


Рис. 12 Кратковременное Дискретное Преобразование Фурье

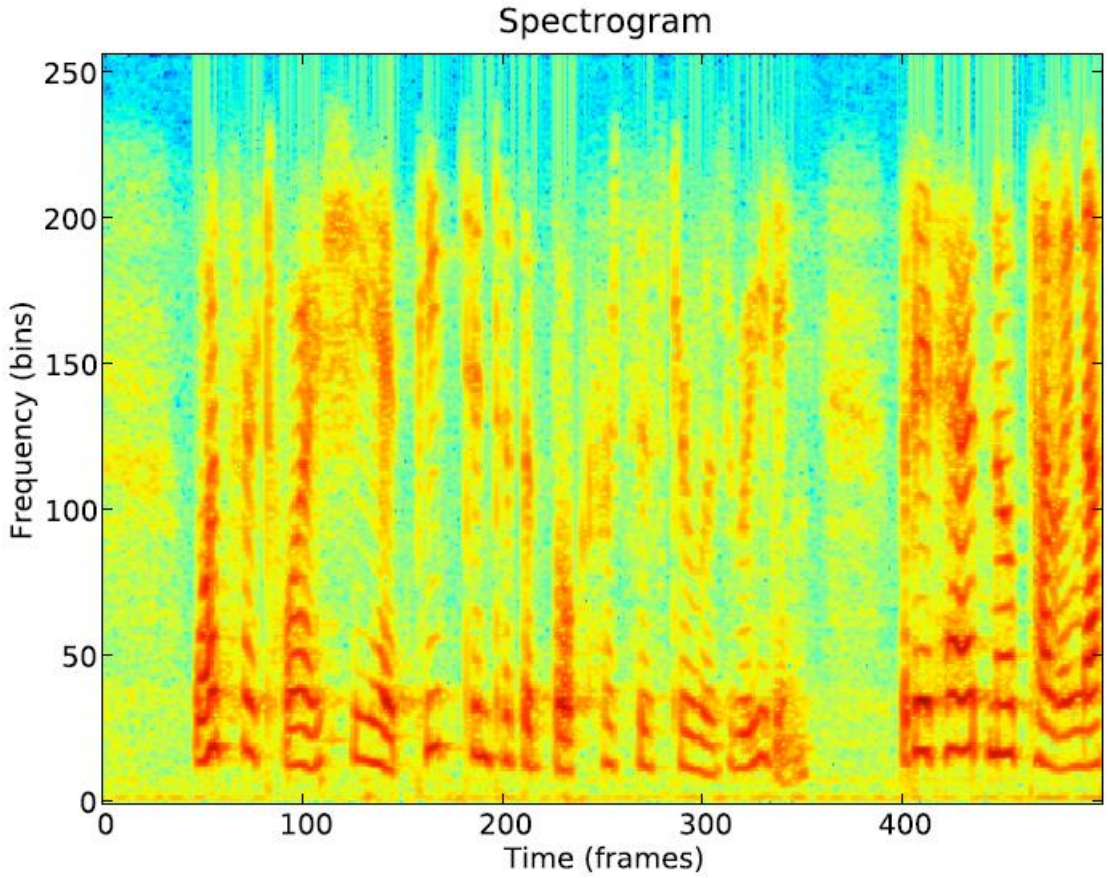
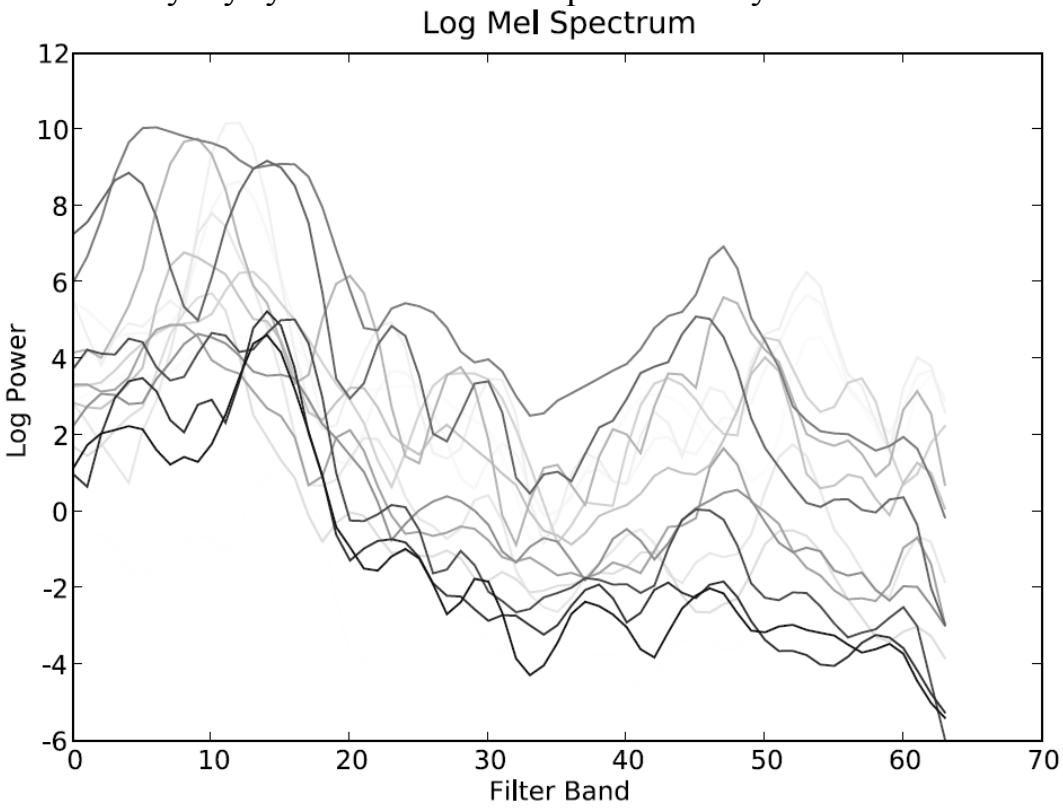


Рис. 13 Спектрограмма – зависимость СПМ от времени

5. СПМ отображается на Mel шкалу (логарифмическая частотная шкала) используя набор фильтров логарифмически-линейно распложенных друг от друга и берется логарифм откликов фильтра. Это трансформирование производит сигнал который логарифмически линеен к изменениям в амплитуде и частоте звука, и смоделирован по “закону мощности” отклика внутреннего уха человека. Набор фильтров использованных в экспериментах является коллекцией 64ех треугольных фильтров, расположенных логарифмически-линейно по ширине оси частот, и с шириной пропускания выбранной так, чтобы фильтры значительно перекрывались и покрывали частоты от 330 Hz до 3300 Hz(выбранных как худший случай для анализа музыки). Рисунки ниже показывает результат логарифмически линейного Mel спектра как индивидуальных спектров (Рис. 14) и как спектрограмму (уложенных во времени, как изображение) (Рис. 15). Эти рисунки представляют тот же сигнал что отображен в шаге 4, то есть может быть напрямую сравнен с начальной СПМ выше.

Логарифмический Mel спектр использован как входной сигнал на HTM во всех экспериментах для анализа музыки.

В данной работе, логарифмический Mel спектр - это окна, с частотой 100 на секунду звука и 64 частотными карманами на окно. Такая структура свойств инварианта к частоте, в значительной степени к узкополосному шуму и изменениям в громкости музыки.



Более светлые линии являются спектром от более старых окон.

Рис. 14 Логарифмический Mel спектр для окон

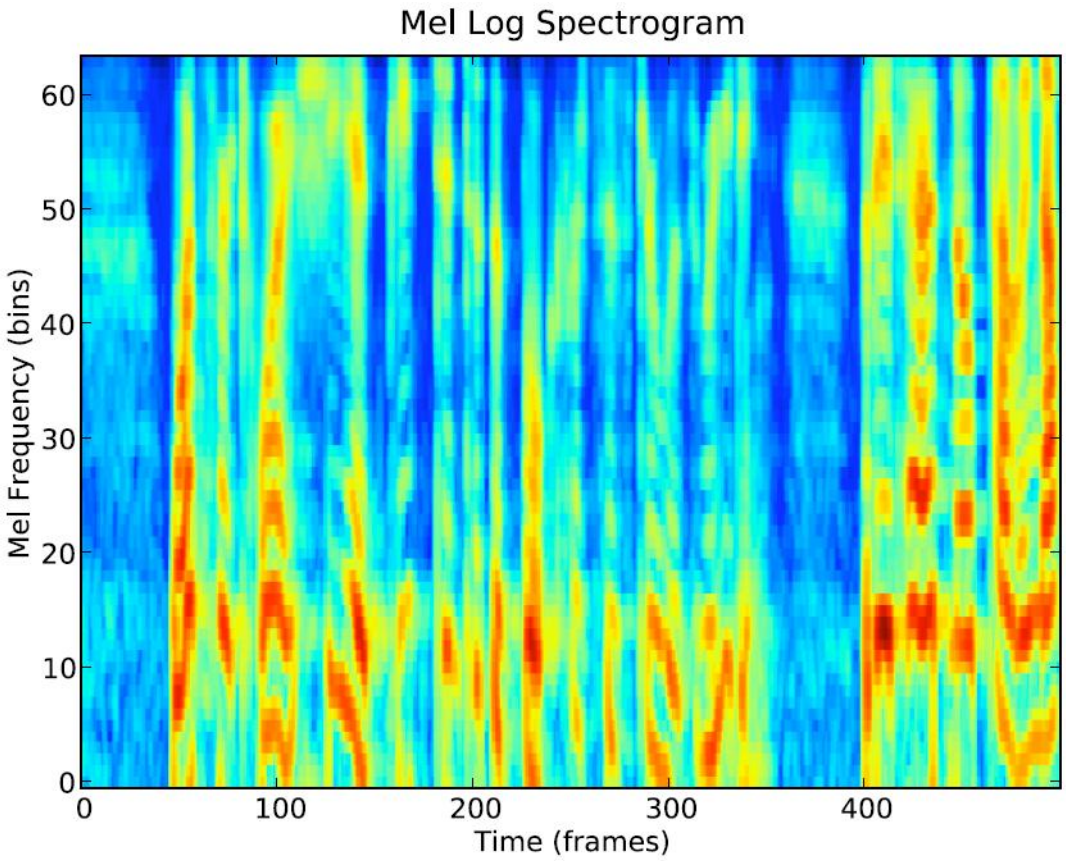


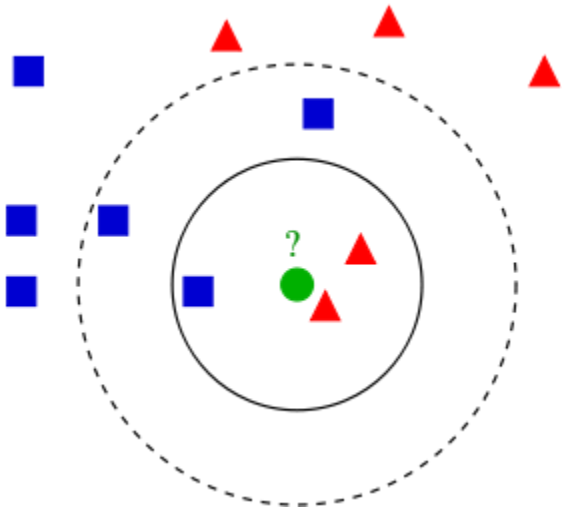
Рис. 15 Спектрограмма Mel спектра

3.2 Алгоритм классификации

Для классификация полученных векторов признаков был использован алгоритм KNN.

KNN - метод автоматической классификации объектов. Основным принципом которого является то, что объект присваивается тому классу, который является наиболее распространённым среди соседей данного элемента.

Соседи берутся исходя из множества объектов, классы которых уже известны, и, исходя из ключевого для данного метода значения k высчитывается, какой класс наиболее многочислен среди них. Визуальный пример можно видеть на Рис. 16



Тестовый образец (зеленый круг) должен быть классифицирован как синий квадрат (класс 1) или как красный треугольник (класс 2). Если k = 3, то она классифицируется как 2ой класс, потому что внутри меньшего круга 2 треугольника и только 1 квадрат. Если k = 5, то он будет классифицирован как 1ый класс (3 квадрата против 2ух треугольников внутри большего круга).

Рис. 16 Пример KNN классификации.

Это методы был использован в силу своей простоты и потому что в данной работе ведется оценка НТМ, а не классификатора. Подобным образом были проведены эксперименты в [16] и [17].

4 ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ

В данной части работы описываются анализируемые данные, последовательность методов предобработки и анализа, результаты проведенного эксперимента.

4.1 Последовательность и параметры обработки и анализа

За основы были приняты стандартные методы и параметры, используемые в подобных экспериментах [18] [4] [16].

Далее описаны шаги, которые были сделаны при проведении эксперимента[4]:

1) Данные
Было выбрано[5] 10 исполнителей, из различных произведений которых было взято по 10 отрывков по 5 секунд каждого автора. 5 исполнителей использовали вокал в своих композициях.

2) Предобработка
Вначале был получена оценка спектральной плотности мощности, из неё были получены MFCCs.

3) Обучение HTM
Использовалась одноуровневая HTM сеть из 4ех узлов. Попытки построить многоуровневые сети не дали хороших результатов. HTM была обучена на 4 отрывках из различных произведений каждого автора.

4) Классификация
Гипотезы выдвигаемые HTM были использованы для обучения KNN[6] классификатора. Тестирование проводилось на 6 отрывках, непересекающихся с тренировочным множеством) по 5 секунд. KNN также был обучен на Mel спектре без использования HTM для получения результатов для сравнения.

---

[4] Параметры сети и модуля предобработки указаны в Приложение А. Инструментарий и программная реализация.
[5] Музыка была взята из базы http://tagatune.org/Magnatagatune.html, которая специально была создана для проведения MIR исследований
[6] Использовалось k=1 и Евклидова метрика.

4.2 Результаты

Далее приведена матрица ошибок идентификации исполнителя с использованием HTM:

Max:

4 0 0 0 0 0 0 0 0 0
0 4 1 0 0 0 0 1 0 0
0 0 3 0 0 0 0 0 0 1
0 0 0 5 0 0 0 0 0 0
1 0 2 1 6 0 0 2 1 2
0 0 0 0 0 6 0 0 2 0
0 0 0 0 0 0 4 0 0 0
1 2 0 0 0 0 2 3 0 0
0 0 0 0 0 0 0 0 3 0
0 0 0 0 0 0 0 0 0 3

Accuracy: 41.0 60.0 = 68.3333333333 %

И без:

Max:

6 0 1 0 0 0 0 2 0 1
0 5 0 0 0 0 0 1 0 0
0 1 1 0 0 0 0 0 0 0
0 0 0 5 0 0 0 0 0 0
0 0 1 1 6 0 0 0 0 1
0 0 0 0 0 6 1 2 3 0
0 0 2 0 0 0 4 0 0 2
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 3 0
0 0 1 0 0 0 0 0 0 2

Accuracy: 38.0 60.0 = 63.3333333333 %

Угадывание в обоих случаях дает 10% точность.

Из приведенных выше данных можно видеть, что после HTM идентификация исполнителей на 5% лучше, чем без неё на 5ти секундных отрывках.

А на 30 секундных отрывках HTM дает 100% точность идентификации, в то время как без неё получается 70%.

Матрица смешения на наличия вокала при анализе с использованием

HTM представлена ниже[7]:

Max:
23 6
7 24
Accuracy: 47.0 60.0 = 78.3333333333 %

Без HTM:
Max:
20 5
10 25
Accuracy: 45.0 60.0 = 75.0 %

Видно, что после HTM точность определения использования вокала исполнителями на 3% больше.

Полное время обучения и классификации с HTM – 1 час 30 минут, полное время без HTM – 3 часа 20 минут.

Из приведенных данных видно, что HTM уменьшает количество образцов для подаваемых на классификатор, не требует слишком долгого времени для обучения и привносить знания о предметной области для более качественного решения задачи.

[7] Для определения наличия вокала использовалась так же обученная сеть но в режиме TBI(Time Based Infernce), которое использует знание о паттернах пришедших в прошлый момент для оценки того какой паттерн пришел в текущий.

5 ЗАКЛЮЧЕНИЕ

В результате проделанной работы можно сделать следующие выводы:

1) HTM можно использовать для обработки музыки, как промежуточное звено между классификатором и обработанными аудио данными для улучшения скорости и качества классификации.
2) HTM создает модели, позволяющее использовать одну и ту же обученную сеть для двух задач.

Пути и цели дальнейшей работы:

1) Получать значение “тембра” и других моделей из музыкально области как предобработка перед подачей данных на HTM. HTM поддерживает ассиметричную многоуровневую иерархию, что дает обширные возможности для экспериментов.
2) Получать значения “тембра” и других музыкальных моделей для сравнения качества классификации по вычисленным векторам признаков с HTM. Данные могут использованы для понимая того, какую модель строит HTM.
3) Использовать информацию о периодичности и очередности паттернов, а не только временную близость. Текущая доступная реализация частично поддерживает работу с очередями [19], а также на перспективы такого подхода явно указывают исследования [9].
3) Использовать оптимизацию для автоматического подбора параметров HTM сети.
4) Использовать n-разовую непересекающуюся случайную выборку звуковых данных для получения качественных статистических данных. Также использовать для анализа задачи и цели из MIREX[8] для сравнения с специально построенными моделями для различных конкретных задач MIR.

[8] http://www.music-ir.org/mirex/wiki/Main_Page The Music Information Retrieval Evaluation eXchange (MIREX) является ежегодной компанией по оценке алгоритмов MIR.

6 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Lillie, A.: MusicBox: Navigating the space of your music., Massachusetts Institute of Technology (September 2008)
2. Jehan, T.: Creating Music by Listening., MASSACHUSETTS INSTITUTE OF TECHNOLOGY (2005)
3. Lartillot, O.: MIRtoolbox 1.3 User's Manual.
4. Tzanetakis, G.: Manipulation, analysis and retrieval systems for audio signals. DISSERTATION (2002)
5. Downie, J.: Music information retrieval (Chapter 7). Annual Review of Information Science and Technology(37), 295-340 (2003) Available from http://music-ir.org/downie_mir_arist37.pdf.
6. George, D., Jaros, B.: The HTM Learning Algorithms.
7. Jeff Hawkins, .: On Intelligence 1st edn. Times Books, New York: Henry Holt (2004)
8. Schey, N.: SONG IDENTIFICATION USING THE NUMENTA PLATFORM FOR INTELLIGENT COMPUTING. A Bachelors of Science Honors Thesis, The Ohio State University (2008)
9. James B. Maxwell, P.: Hierarchical Sequential Memory for Music: A Cognitive Model. (2009)
10. Pampalk, E.: Islands of Music. Analysis, Organization, and Visualization of Music Archives. DIPLOMARBEIT (December 2001)
11. George, D.: How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition. PhD Thesis, Stanford University, Palo Alto, CA (2008)
12. Ali Nouri, H.: Hierarchical Bayesian Reservoir Memory., Bu-Ali Sina University, Hamedan, Iran (2009)
13. Jim Mutch, U.: CNS: a GPU-based framework for simulating cortically-organized networks., McGovern Institute for Brain Research Massachusetts Institute of Technology Cambridge, MA (2010)
14. HawkinsJ., George, D.: A Hierarchical Bayesian Model of Invariant Pattern Recognition in the Visual Cortex., Redwood Neuroscience Institute, Menlo Park, CA (2004)
15. Jim Mutch, D.: Object class recognition and localization using sparse features with limited receptive fields. (2008)
16. Beth, L.: Nearest-Neighbor Artist Identification., Cambridge MA USA (2004)
17. Robinson, D., Leung, K., Falco, X.: Spoken Language Identification With Hierarchical Temporal Memories. (2009)
18. Inc., Numenta: Speech Processing with Hierarchical Temporal Memory.
19. Inc., N.: Node Algorithms Guide. (2008)
20. Kirss, P.: Audio Based Genre Classification of Electronic Music. Master's Thesis, Music, Mind and Technology University of Jyväskylä (2007)
21. Inc., Numenta: Getting Started With NuPIC.
22. Dileep George, J.: Hierarchical Temporal Memory: Concepts, Theory, and Terminology. (2007)

23. Joost van Doremalen, L.: Spoken Digit Recognition using a Hierarchical Temporal Memory., University of Nijmegen, the Netherlands (2008)
24. Lartillot, O., Toiviainen, P.: A MATLAB TOOLBOX FOR MUSICAL FEATURE EXTRACTION FROM AUDIO. Proc. of the 10th Int. Conference on Digital Audio Effects (DAFx-07), Bordeaux, France (2007)

Приложение А. Инструментарий и программная реализация

(справочное)

Все эксперименты производились лэптопе с 32ухбитной Windows 7 с процессором Intel Core 2 Duo 1.83 ГГц и 2ГБ DDR2 оперативной памяти.

Данные для анализа были собраны из звуковых файлов, используя MATLAB 2009B и MIRToolbox 1.3[9].

```
function readMP3ConcatSaveWAV(parentfolder, frequency, nSeconds)
%READMP3CONCATSAVEWAV  Reads and concatenates MP3 files from each subfolder of
parentfolder.
%   Creates file exactly matching subfolder name with WAV extension.
% PARENTFOLDER Folder with subfolders containting MP3 files.
%   Files in single subfolder are targets of concatenation.
% FREQUENCY Sampling frequency
% NSECONDES Hove much seconds of each file to read from start.
mirverbose(0)
mirwaitbar(0)
%mirerror(0)
folders = dir(parentfolder);
minimalFilesCount = 100;
step = 7;
% skipping '.' and '..' dirs
for nFolder=3:length(folders)
   if ~(folders(nFolder).isdir)
      continue;
   end
   currentFolderName = folders(nFolder).name;
   currentFolderPath = [parentfolder '/' folders(nFolder).name];
   files = dir([currentFolderPath '/*.mp3']);
   if (isempty(files))
      continue;
   end
  a = [];
   % concatenating
   for nFiles=1:step:100
      % read another file if first could not be read
      for tries=0:step-1
         try
            currentFilePath = [currentFolderPath '/' files(nFiles+tries).name];
            a = [a ; mirgetdata(specialmiraudio(currentFilePath,nSeconds))];
            continue;
         catch exception
            % some files can't be read
         end
      end
   end
   saveAs = [parentfolder '/' currentFolderName];
   wavwrite(a,frequency,16,saveAs);
end
```

[9] https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

```
    function audio = specialmiraudio(filePath,nSeconds)
        audio = miraudio(filePath,'Sampling',frequency,'Extract',0,nSeconds);
    end
end
```

Листинг 1 Чтение и конкатенация аудио файлов

Для проведения экспериментов был использован Nupic 1.7.1[10] (закрытая реализация HTM), частью которого является Speech Toolkit основанный на библиотеках с открытым исходным кодом для анализа аудио данных. Используется язык Python.

Далее приведены листинги настроек экспериментов использованных в данной работе, которые основаны на [18] .

```
from nupic.support.speech import *

network = Network([
    ("sensor", load(ExperimentPath("sensor.py"), selection="sensor")),

    Region(
        name="level1",
        layout=1024,
        nodeType="PassThroughNode",
        phase=2,
        dataOut = 1,
      ), # End of level1.
    Link(
        linkType="SensorLink",
        direction="up",
        source="sensor",
        sourceOutputName="dataOut",
        overlap=[0],
        destination="level1",
      ),
  ]) # End of Network.

train = Program(
    SetParameter("sensor", "sequences", ExperimentPath("data/train")),

    Save(ExperimentPath("networks/trained.xml")),
    # Reload from the checkpoint.
    Load(ExperimentPath("networks/trained.xml")),

    Classify(

        source="level1",
        sourceOutput="dataOut",
        categorySource="sensor",
        categorySourceOutput="categoryOut",
        nodeType="py.KNNClassifierNode",
        categoriesOut=11, # 1-10
```

[10] http://numenta.com/for-developers.php

```
    trainingProgram=Program(
      SetParameter("sensor", "sequences", ExperimentPath("data/train")),
      Run("classifiers.level1.classifier",
        TrainPhase("classifiers.level1.classifier",
          ForEachSequence(["sensor"], 0.25))
      ),
    ),

    testPrograms=dict(
      quick=Program(
        SetParameter("sensor", "sequences", ExperimentPath("data/quick")),
        Run("classifiers.level1.analyzer", ForEachSequence(["sensor"], 0.25)),
      ),
      test=Program(
        SetParameter("sensor", "sequences", ExperimentPath("data/test")),
        Run("classifiers.level1.analyzer", ForEachSequence(["sensor"], 0.25)),
      ),
    ),
  ),
)

experiment = DefineExperiment()
```

Листинг 2 Сеть без использования HTM.

```
from nupic.support.speech import *

network = Network([
  ("sensor", load(ExperimentPath("sensor.py"), selection="sensor")),

  Region(
    name="level1",
    layout=4,
    nodeType="Zeta1Node",
    phase=1,
    spatialPoolerAlgorithm="gaussian",
    detectBlanks=      0,
    sigma=          4.0,
    maxDistance=    32.0,
    transitionMemory=  1,
    symmetricTime=     1,
    topNeighbors=      3,
    maxGroupSize=     16,
    overlappingGroups= 1,
    bottomUpOut=    1024,
    temporalPoolerAlgorithm="sumProp",
  Link(
    linkType="SensorLink",
    direction="up",
    source="sensor",
    sourceOutputName="dataOut",
    overlap=[0],
    destination="level1",
  ),
]) # End of Network.

train = Program(
  SetParameter("sensor", "sequences", ExperimentPath("data/train")),
  Run(selection="level1",
      policy=TrainPhase("level1", ForEachSequence(["sensor"], 0.25))),
```

```
    Save(ExperimentPath("networks/trained.xml")),
    # Reload from the checkpoint.
    Load(ExperimentPath("networks/trained.xml")),

    Classify(

      source="level1",
      sourceOutput="bottomUpOut",
      categorySource="sensor",
      categorySourceOutput="categoryOut",
      nodeType="py.KNNClassifierNode",
      categoriesOut=11, # 1-10

      trainingProgram=Program(
        SetParameter("sensor", "sequences", ExperimentPath("data/train")),
        Run("classifiers.level1.classifier",
          TrainPhase("classifiers.level1.classifier",
            ForEachSequence(["sensor"], 0.25))
        ),
      ),

      testPrograms=dict(
        quick=Program(
          SetParameter("sensor", "sequences", ExperimentPath("data/quick")),
          Run("classifiers.level1.analyzer", ForEachSequence(["sensor"], 0.25)),
        ),
        test=Program(
          SetParameter("sensor", "sequences", ExperimentPath("data/test")),
          Run("classifiers.level1.analyzer", ForEachSequence(["sensor"], 0.25)),
        ),
      ),
    ),
  )

experiment = DefineExperiment()
```

Листинг 3 Zeta1 уровень из 4ух узлов

```
k=1,
distanceNorm=2.0,
distThreshold=0,
doBinarization=False,
inputThresh=0.500,
useSparseMemory=True,
sparseThreshold=0.0,
relativeThreshold=False,
numWinners=0,
acceptanceProbability=1.0,
seed=None,
doSphering=False,
numSVDSamples=None,
numSVDDims=None,
fractionOfMax=None,
```

Листинг 4 Параметрами инициализации используемого KNN классификатора