Treball de Fi de Carrera (TFC).

Implementación y entrenamiento de un modelo clasificatorio de red neural sobre la base de datos IGBADAT para la clasificación de las rocas basálticas de acuerdo a las clases del sistema de clasificación tradicional de Yoder and Tiller.

Nom Estudiant: **David Tramuns Monterde (ETIS)**

Nom Consultor: **Raimon Caihuelas Quiles**

Data Lliurament: 30/12/2004

RESUMEN DEL PROYECTO.

En los últimos años las técnicas de ***data mining*** han sido progresivamente incorporadas a los métodos de estudio de la geología y las ciencias de la Tierra, en parte debido al gran volumen de datos del cual se dispone y que requiere un notable esfuerzo de análisis. Sin embargo, su uso no se ha extendido por igual en todas las especialidades geológicas, siendo las áreas que tradicionalmente hacen un mayor uso de los métodos numéricos, tales como la geofísica, la hidrogeología y la teledetección, donde éste se halla más extendido, mientras que en las áreas más cercanas a la geología clásica, más habituadas a trabajar con datos descriptivos o "blandos" es mucho menor. Asimismo, muchas de las aplicaciones hasta ahora consisten en implementar modelos predictivos de variables geológicas mediante el uso de redes neurales de retropropagación.

En el presente trabajo se pretende implementar un proyecto de ***data mining*** en el área de la petrología ígnea, especialidad englobada dentro de la geología clásica. Dentro del conjunto de las rocas ígneas, los basaltos son un importante grupo de rocas que tradicionalmente han sido objeto de clasificación atendiendo a diversos criterios: clasificaciones de acuerdo a mineralogía normativa, clasificaciones modales, clasificaciones químicas y/o geoquímicas, clasificaciones de acuerdo a criterios petrográficos simples,...y con frecuencia los diferentes criterios de clasificación llevan a denominaciones similares.

El objetivo del proyecto es comprobar si se puede entrenar un modelo clasificador de red neural mediante el paquete de programas WEKA que relacione la composición analítica de las rocas con las denominaciones del modelo de clasificación normativo de Yoder and Tiller , seleccionando los registros de la base de datos de petrología ígnea de la IUGS que corresponden a las tres denominaciones principales de las clases de este sistema, comprobando además si éstas tienen la suficiente coherencia a pesar de las múltiples fuentes posibles de origen de la denominación.

INDICE DEL PROYECTO.

1 INTRODUCCIÓN: TÉCNICAS DE DATA MINING EN GEOLOGIA.

Los avances en las tecnologías informáticas en los últimos años ha tenido como consecuencia de que muchas áreas científicas y de ingeniería dispongan de volúmenes masivos de datos obtenidos tanto a partir de procesos de simulación, como de la observación y experimentación. Un simulación informática puede generar en pocas horas volúmenes de datos del orden del Terabyte , lo cual hace necesario el desarrollo de herramientas y técnicas que haga posible a los analistas humanos la obtención de información útil de los mismos, entre los que empieza ha ser objeto de importante consideración las técnicas de minería de datos . Kamath (2001) efectúa una revisión de las particularidades en la aplicación de técnicas de ***data mining*** sobre datos científicos. El autor cita tres tipos de categorías de datos diferentes que pueden hallarse:

- Datos unidimensionales: típicamente registros recogidos por un sensor, con frecuencia correspondiente a una serie temporal *(por ejemplo, datos de precipitación recogidos por un pluviómetro en una estación meteorológica a lo largo de un mes)*

- Datos bidimensionales: dos casos típicos pueden ser una imagen obtenida a partir de una fotografía aérea o de satélite o bien una simulación informática bidimensional que puede, por ejemplo, representar una evolución temporal de un sistema

- Datos tridimensionales: típicamente obtenidos a partir de una simulación en tres dimensiones, en las cuales también se puede tener en consideración tanto aspectos espaciales como de evolución temporal.

Este autor destaca algunas de la áreas científicas en las que puede ser de mayor utilidad la aplicación de técnicas de ***data mining***:

- **Astronomía:** con la nueva generación de telescopios , detectores y cámaras CCD para fotografía astronómica se disponen de conjuntos de datos almacenados que fácilmente alcanzan el Terabyte, correspondientes a imágenes digitales o series de datos con millones de registros , a veces con cientos de atributos. Como ejemplo de algunas aplicaciones en esta área usando técnicas de ***data mining*** son la clasificación de objetos difusos, detección de volcanes en Venus, clasificación de estrellas y galaxias,...

- **Biología , química y medicina:** la bioinformática dispone y analiza grandes volúmenes de datos de secuencia genéticas y proteínas. Algunos ejemplos de aplicaciones serían la identificación de genes , secuencias de ADN, automatización en técnicas de cristalografía de proteínas para la estructura de las mismas, análisis de imágenes (mamografías, ultrasonidos, rayos X,...) para identificar patologías, análisis de datos obtenidos en simulación en el área de química computacional,...

- **Ciencias de la Tierra:** se dispone de gran volúmenes de datos de simulación climática y atmosférica , imágenes digitales obtenidas por teledetección y por el desarrollo de sistemas de información geográfica (GIS). Algunas aplicaciones son clasificación automática de objetos a partir de datos obtenidos mediante teledetección, investigación de las causas de la desaparición de la capa de ozono y el aumento del efecto invernadero, gestión del territorio, detección de terremotos desde el espacio, entendimiento de la interacciones entre atmósfera, biosfera, geosfera e hidrosfera,...

Kamath (2001) cita también alguna particularidades y diferencias respecto a datos y bases de datos de origen comercial y de gestión empresarial que debe tenerse en cuenta en el momento de plantearse proyectos de minería de datos científicos:

- Los datos presentan error experimental (*Noisy data*) que puede ser variable. La determinación y eliminación en lo posible del error en los datos sin afectar la señal analizada debe ser un elemento a tener en cuenta. Asimismo el uso de valores de substitución en un atributo, cuando éste está ausente en un registro de la serie de datos o bien es claramente erróneo, debe de tener una clara justificación científica.

- En el tratamiento previo de datos no debe de perderse de vista que éstos con frecuencia representan una realidad física real.

- Las series de datos pueden ser de tamaño moderado e incluso masivos, alcanzando tamaños del orden del Terabyte, y se espera que aumente aún más en los próximos años.

- Los datos pueden haber sido obtenidos a partir de múltiples fuentes distintas, lo cual hace necesario el uso de técnica de fusión de datos para lograr un formato de archivo que permita aplicar las técnicas de minería. Hay que tener presente que los datos pueden haber sido tomados con diferentes sensores , resoluciones y condiciones.

- Determinación del valor de los datos categóricos: no todos los científicos pueden estar de acuerdo en el valor que debe tomar un mismo atributo categórico en un mismo objeto, cuando se actúa con criterios de asignación subjetivos[1]. Esto debe de ser tenido en cuenta en las series de datos de validación que se han generado con frecuencia manualmente y/o a partir de diversas fuentes y autores. Peor aún, en algunos campos, como la astronomía, se puede tener dificultades para obtener una serie de registros para llevar a cabo la validación del modelo que sean totalmente fiables, al no poderse obtener directamente a partir de datos de laboratorio y no tener acceso directo físico a los objetos de estudio.

[1] Esto se tendrá especialmente presente en este trabajo

- Los datos suelen estar disponibles en un solo archivo o unos pocos archivos de estructura lógica simple y raramente se han diseñado para constituir una base de datos compleja.

- Con frecuencia, se necesita que los resultados del ***data mining*** sean obtenidos rápidamente para integrarlos en procesos en tiempo real o bien ejecutar la minería de datos inmediatamente a la generación de los mismos, por ejemplo para corregir resultados obtenidos en cada paso de una simulación temporal integrando la técnica de minería en la propia simulación. Se debe tener presente en la integración de la minería que el volumen de la base de datos puede crecer significativamente.

- El trabajo de minería de datos científicos no suelen presentar las complicaciones de temas de seguridad, privacidad y/o propiedad que tiene los datos comerciales. En muchos casos son públicos y gratuitos. Sin embargo, la minería puede ser poco provechosa si no se efectúa en estrecha colaboración con científicos del área para entender los mismos e identificar problemas relevantes

La geología constituye actualmente una ciencia englobada dentro de las Ciencias de la Tierra. Una especialidad de la misma es la petrología ígnea, que estudia la mineralogía, petrografía, composición de las rocas de origen magmático, así como el origen de las mismas, de los magmas que las originan y también su clasificación.

Históricamente las ciencia geológica clásica a sido una disciplina más de metodología empírica y observacional que experimental, por lo que buena parte de la información disponible es de tipo descriptivo[2], quedando la metodología experimental para obtener datos numéricos[3] y el uso de herramientas físico-matemáticas en el análisis de los mismos en un segundo plano en muchas subdisciplinas de la misma . Sin embargo en las últimas décadas han adquirido un rápido desarrollo determinadas áreas con una mayor utilización de metodología experimental y de modelización físico-matemática de objetos y procesos geológicos, disponiéndose actualmente de un importante volumen de datos numéricos. En los últimos años se ha puesto especial interés en la construcción de modelos clasificatorios y predictivos usando redes neurales de retropropagación. (BPNN)[4]. Por su parte, Lees (1996) advertía que las redes neurales pueden ser útiles en problemas que violan las precondiciones fundamentales que se establecen en los métodos de análisis tradicionales . Así, por ejemplo, el análisis integrado de bases de datos espaciales, ambientales, temporales,...cuyas relaciones entre atributos son complejas , se realiza por métodos paramétricos tradicionales, mediante el establecimiento de asunciones previas de difícil cumplimiento en la realidad. El uso de redes neurales no requiere tales asunciones.

[2] En argot de los autores que trabajan el tema, se dice que usan datos “blandos”
[3] Análogamente, se dice que usan datos “duros”
[4] Abreviatura de **B**ack **P**ropagation **N**eural **N**etwork

Más genéricamente, Masters (1993) propone que los modelos matemáticos construidos a partir de redes neurales serán probablemente superiores a otros métodos cuando se cumplen las siguientes condiciones:

a) Los datos a partir de los cuales se construye el modelo corresponden a valores de categorías mal definidas, o son producto de la opinión humana o simplemente pueden estar sujetos a grandes errores.

b) El patrón de comportamiento subyacente a los datos es muy sutil o bien se halla oculto. Las redes neurales pueden descubrir patrones de comportamiento en datos que son imperceptibles tanto para investigadores humanos como a métodos estadísticos clásicos.

c) Los datos exhiben un comportamiento de no-linearidad impredecible. Las redes neurales son mucho más adaptables que los modelos basados en un comportamiento definido de los mismos, los cuales serán de poca utilidad cuando su comportamiento real se aleje del mismo.

d) Los datos tienen un importante comportamiento caótico. Los modelos basados en redes neurales tienen un comportamiento más robusto que los construidos usando otras técnicas en estas circunstancias.

Las variables geológicas presentan con frecuencia este comportamiento, explicando el importante esfuerzo de la comunidad científica geológica en investigar y desarrollar posibles aplicaciones de los modelos basados en BPNN.

Para ilustrar esto último, realizando una simple búsqueda en la base de datos de publicaciones geológicas GEOREF[5], usando como parámetro de búsqueda las palabras “neural network” , se puede encontrar un total de 723 artículos sobre el tema[6]. En la figura 1 se observa como no existían referencias hasta el año 1986. A partir de esta fecha , el igual que en otras disciplinas científicas [Gurney,1997], nace el interés en el estudio de aplicaciones de la BPNN, aumentando notablemente a mediados de los años noventa hasta la actualidad.

[5] GEOREF , base de datos que recoge todas las referencias de publicaciones de artículos y libros conocidas desde 1785 de temática geológica. Se puede acceder libremente a la misma desde las bibliotecas de la Universidad de Barcelona.

[6] Dato actualizado para Agosto del 2004; en GEOREF se van añadiendo nuevas referencias continuamente

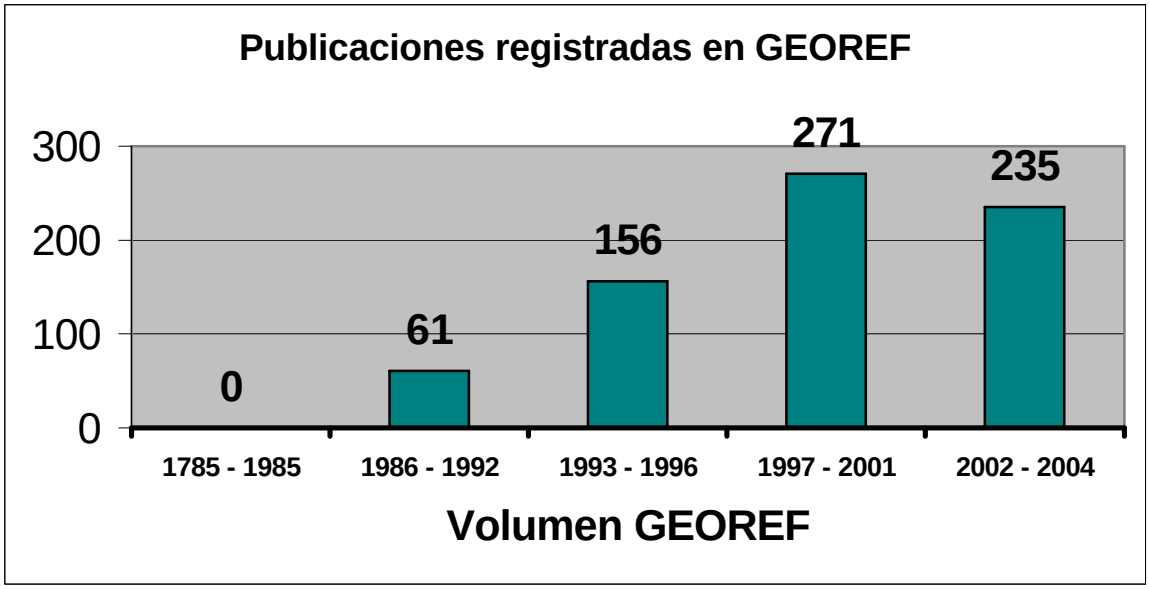

Figura 1.1 Resultado de la búsqueda en la BBDD Georef sobre aplicaciones geológicas de las redes neurales, por volúmenes de la misma, hasta agosto 2004.

En la Tabla 1.1 puede verse el número de artículos encontrados clasificados según periodos y áreas temáticas. Se observa como los modelos basados en BPNN fueron usados en investigación geofísica, pero a mediados de los años 90 la investigación en Geología del Petróleo, Hidrología, Geotecnia entre otros , también se interesa de modo creciente en estos modelos. Algunas de éstas disciplinas son las que tradicionalmente más han usado la metodología experimental, implicando la medida de parámetros físicos y recogida de valores numéricos, así como el desarrollo de modelos matemáticos a partir de los mismos, mientras que en las disciplinas de "geología clásica" , que dan mayor peso a la recogida de información descriptiva, el uso de los modelos BPNN es todavía testimonial.

En el presente proyecto se intenta recoger algunas ideas de estos autores y llevar a cabo una aplicación de minería de datos sobre una base de datos pública como es la base de datos de petrología ígnea de la IGBA, de cara intentar reproducir algunos esquemas clásicos de clasificación de rocas basálticas. Por consiguiente , se intenta presentar un proyecto de minería de datos en una disciplina de la geología clásica de la que se dispone bastante información de datos "duros" geoquímicos, pero que tradicionalmente se han utilizado relativamente poco en técnicas físico-matemáticas de modelización y análisis.

Area	1986 - 1992	1993 – 1996	1997 - 2001	2002 – 2004
Geofísica, sismología y Exploración Geofísisca	38	42	44	46
Geología del Petroleo	6	33	54	33
Geología Minera, Exploración y Yacimentos Minerales	2	5	16	18
Geoquímica, Geoquímica y Medioambiente	2	1	6	7
Geotecnia, Petrofísica y Sondeos	3	13	30	32
Hidrología y Hidrogeología	3	8	36	21
Petrología Sedimentaria	2	5	4	1
Teledetección, GIS y Cartografía	3	23	16	16
Mineralogía y Cristalografía	2	1	1	0
Geomorfología, Geodinámica Externa Y Riesgos Geológicos	0	4	19	18
Estratigrafía y Sedimentología	0	6	13	14
Geocomputación y Geomatemáticas	0	13	22	22
Geoplanetología	0	2	0	0
Paleontología	0	0	3	4
Climatología y Paleoclimatología	0	0	6	1
Vulcanología	0	0	1	2

Tabla 1.1 Clasificación temática según áreas de aplicación de las citas de publicaciones geológicas sobre redes neurales encontradas en Georef. En algunos casos la atribución a un tema es algo arbitraria, pues una misma publicación podría abarcar simultáneamente más de un área

2 OBJETIVOS DEL PROYECTO.

El objetivo del TFC es explorar la posibilidad de que se pueda usar técnicas de *data mining* en bases de datos petrológicas, que resumen información geoquímica, mineralógica y textural de las muestras de rocas ígneas publicadas en revistas científicas de contenido geológico. Se explora la coherencia de la clasificación de las mismas en la base de datos usando técnicas y criterios clásicos en petrología comparándose con los resultados obtenidos a partir de modelos clasificatorios con redes neurales y la aportación de los mismos en la obtención de información petrológica significativa.

Se usa la base de datos de petrología ígnea IGBADAT para seleccionar los registros utilizados en el análisis y sus atributos, intentando reproducir el esquema **clasificatorio normativo** clásico mediante gráficos de **YODER and TILLER** para las rocas basálticas a partir de su composición químicas de sus componentes mayoritarios, expresada en formad óxidos, utilizando un modelo de red neural. Se procederá de acuerdo con las siguientes fases:

a) Obtención de la base de datos IGBADAT y análisis de la misma.

b) Tratamiento de datos: selección de registros de rocas basálticas, correspondientes a los descritos como **basalto olivínico, basalto alcalino y toleita** (términos de la clasificación de YODER and TILLER), y sus respectivos atributos de análisis de óxidos químicos mayoritarios, para obtener un fichero ARFF que se usará en la construcción del modelo mediante el paquete de software WEKA

c) Reproducción de los esquemas clasificatorios de los diagramas de YODER and TILLER usando un modelo de red neural de retropropagación aplicado a datos de análisis químico de los elemento mayoritarios de las rocas.

d) Búsqueda de la mejor arquitectura del modelo.

e) Resultados y discusión . Conclusiones: evaluación del modelo obtenido y discusión del problemas de solapamiento composicional de los datos de IGBADAT.

3 ASPECTOS MATEMÁTICOS Y COMPUTACIONALES DE LOS MODELOS CLASIFICATORIOS BASADOS EN REDES NEURALES DE RETROPROPAGACIÓN (BPNN).

Al igual que en otros tipos de modelos clasificatorios usados en *data mining*, la construcción del modelo puede efectuarse a partir de un conjunto de N registros de la base de datos, de los cuales se pretende seleccionar un cierto conjunto vectorial de atributos $\mathbf{x}$[7] , a partir de los cuales se determina uno o más atributos **y,** es decir, se pretende encontrar la mejor función M que cumpla para los registros :$\mathbf{x_i}$ del conjunto de entrenamiento:

$$\mathbf{y_i} = M(\mathbf{x_i})$$

Para entrenar el modelo de red neural M se usará una fracción de los valores $\mathbf{x_i}$ disponibles, **(conjunto de entrenamiento)** pudiendo usarse el resto para evaluar el modelo, **(conjunto de evaluación)**. También puede ser interesante seleccionar una parte de los mismos los mismos para construir un **conjunto de validación** del modelo, una vez construido éste.

El nodo básico de la red es la **neurona**. Ésta puede considerarse como un objeto matemático, que simula el comportamiento de las neuronas cerebrales, con las siguientes características:

- Cada neurona tiene *n* entradas de datos (**input**) , pero un único valor de salida (**output**)
- El valor de salida se determina calculando un valor de activación *a* partiendo de los *n* valores de **input**, y aplicando al mismo una función de activación *f*, de modo que el valor de output *O* para esta neurona sería:

$$O = f(a)$$

En la práctica , un método de obtención del valor de salida se obtiene efectuando los siguientes cálculos: [Masters, 1993]

1) La entrada de *n* número de datos en una neurona puede representarse con un vector de *n* dimensiones $\mathbf{x} = (x_0,x_1,...,x_{n-1})$, y se interpreta cada uno como una conexión sináptica de entrada. La neurona tienen también asociado un vector de *n+1* pesos $\mathbf{w} = (w_0,w_1,...w_n)$. La activación *a* de la neurona se determina como la suma ponderada con los de los valores de entrada:

$$a = \sum_{i=0}^{n-1} xiwi + wn$$

[7] El carácter vectorial del parámetro se resalta en negrita en el texto

El término de la suma w_n corresponde al *bias*. El resto de pesos pueden interpretarse como la contribución de cada conexión sináptica de entrada a la activación de la neurona.[Gurney,1997]

2) Se aplica la función de activación *f* a la suma ponderada *a* anterior. Esta función suele ser una función no lineal, para simular el comportamiento de la neuronas del sistema nervioso. Una función muy utilizada es la función logística:

$$\text{Se } f(x) = \frac{1}{1 + e^{-\frac{(x-\theta)}{\rho}}}$$

en la que los parámetros θ y ρ pueden usarse para ajustar la forma exacta deseada de la función. Esta función tienen la ventaja que su derivada puede calcularse fácilmente como:

$$f'(x) = f(x)\cdot(1 - f(x))$$

Se ha comprobado que con frecuencia la forma exacta de la función de activación utilizada influye poco en la eficiencia del modelo. [Masters, 1993]. Para simplificar los cálculos , puede ser suficiente en el programa de cálculo definir una tabla que relacione simplemente un cierto número de intervalos del valor de activación Δ*a* con una imagen de la función *f(*Δ*a)* que se usarán como output**,** haciendo el proceso de cálculo menos computacionalmente exigente.

La red neural del modelo M se construye usando una arquitectura de tres capas[8] diferentes de neuronas: la capa de entrada o **capa de input**, la ***capa oculta*** y la capa de salida o **capa de output.**

Las neuronas de la capa de input tienen una única entrada. Esta entrada corresponde a uno de los atributos de entrada representados por el vector $\mathbf{x}=(x_0,x_1,...,x_{n-1})$, de modo que si se usan *n* atributos la capa de entrada tendrá *n* neuronas. Todos los outputs de las neuronas de esta capa se usan como valores de entrada en cada una de las neuronas de la capa oculta, de modo que cada neurona de la capa oculta establece *n* sinápsis de entrada con las neuronas de la capa anterior, y cada neurona *i*-ésima de la capa oculta tiene un conjunto de *n+1* pesos $\mathbf{w_i} = (w_{i0},w_{i1},w_{i2},...,w_{in-1},w_n)$ que caracterizan estas sinápsis. (más el término del *bias*). Todos los outputs de las neuronas de la capa oculta se usan como valores de entrada en cada una de las neuronas de la capa de salida, de modo que si la capa oculta se construía con *m* neuronas, la neurona de la capa de salida establece *m* sinápsis de entrada con las neuronas de la capa anterior, y cada neurona *i*-ésima de la capa de salida tiene un conjunto de *m+1* pesos, siendo $\mathbf{w_i} = (w_{i0},w_{i1},w_{i2},...,w_{im-1},w_m)$ que caracterizan estas sinápsis. (más el *bias*). Los outputs de las neuronas de la capa de salida determinan el valor de salida $\mathbf{y_i}$ del modelo. La elección del número de neuronas de esta capa viene determinado por el modo exacto de como se construya el modelo M.

La construcción del modelo M es un proceso iterativo de ajuste de los pesos de las neuronas de la capa oculta y la capa de salida. De modo que se consiga que para un conjunto determinado de pesos, el valor de los outputs $\mathbf{O_i}$ de la capa de salida para las entradas $\mathbf{x_i}$ en la ecuación $\mathbf{O_i}=M(\mathbf{x_i})$, se aproximen al objetivo deseado $\mathbf{y_i}=M(\mathbf{x_i})$ dentro

[8] Suele ser suficiente el uso de una única capa oculta. [Masters,1993]

de un margen de error suficientemente pequeño. Cada paso de este proceso iterativo se denomina *época*, y puede describirse considerando las siguientes fases:

I. Se construye un modelo M´ a partir de un conjunto de pesos w_{ij}[9] i se calcula el valor de output **O**i de la red para todas las entradas $\mathbf{x_i}$ del conjunto de entrenamiento, de modo que :

$$\mathbf{O_i}=M'(\mathbf{x_i})$$

En la primera *época*, se inicializa el conjunto de pesos w_{ij} asignándoles valores pequeños distintos de cero obtenidos aleatoriamente.

II. Para conseguir aproximar el modelo M´ al buscado M se debe estimar el error que se comete al adoptar como modelo válido M´. Una técnica para determinar este error puede consistir en, suponiendo que se usa p-valores de entrada en el conjunto de entrenamiento y que existen n-neuronas en la capa de salida, calcular el error E_p se comete al aplicar el modelo a cada registro de entrada *p* , que será:

$$Ep = \frac{1}{n}\sum_{j=0}^{n-1}(ypj - Opj)^2$$

donde y_{pj} representa el valor de output esperado en la neurona *j*-ésima de la capa de salida para el vector de entrada $\mathbf{x_p}$ del conjunto de entrenamiento y O_{ij} representa el valor realmente obtenido en el modelo.

El error total E del modelo se determinará como:

$$E = \frac{1}{p}\sum_{j=0}^{p-1}Ej$$

Para ajustar el modelo se puede suponer que el error total E es una función de los pesos $\mathbf{w_{ij}}$[10]**.** Imaginando que la función escalar E($\mathbf{w_{ij}}$) define una hipersuperficie en el espacio vectorial de los pesos, la función gradiente del error total $\nabla E(wij)$ nos dará, en el espacio vectorial definido por los pesos, la dirección de máxima pendiente de la misma. Así, si se recalcula los pesos como $\mathbf{w_{ij}^{(nuevo)}=w_{ij}^{(anterior)}} + \Delta\mathbf{w_{ij}}$, en donde el término $\Delta\mathbf{w_{ij}}$ tendría la dirección definida por el gradiente, se debería tender a encontrar el punto donde el valor de E($\mathbf{w_{ij}}$) alcanza su mínimo global, punto en el cual el modelo M´ se adoptaría como modelo M válido.

Para evaluar este gradiente deben estimarse las derivadas parciales del error con respecto los pesos. En las neuronas de la capa de salida, las derivadas respecto el peso que relaciona la neurona *i*-ésima de la capa oculta con la neurona *j*-ésima de la capa de salida pueden evaluarse como:

[9] El símbolo w_{ij} puede interpretarse como el peso asociado a la conexión entre la i-ésima neurona de una capa y la j-ésima neurona de la capa anterior

[10] Es decir, de todos los pesos asociados a las respectivas conexiones sinápticas entre neuronas.

$$\frac{\partial E}{\partial wij} = -Oi \cdot f'(aj)(yj - Oj)$$

siendo:

O_i= valor de output de la neurona i de la capa oculta
O_j= valor de output de la neurona j de la capa de salida
a_j = suma ponderada de los valores de entrada de la neurona j de la capa de salida
y_i= valor deseado para el output de la neurona j
f´ = derivada de la función de activación de la neurona j

que puede expresarse como

$$\frac{\partial E}{\partial wij} = -Oi\delta j$$

$$\text{donde } \delta j = f'(aj)(yj - Oj),$$

siendo $f'(a_j)$ fácilmente calculable si se ha usado la función logística como función de activación.

La derivadas parciales respecto de los pesos de la capa oculta, si w_{ij} es el peso que relaciona la *j*-ésima neurona de la capa oculta con la *i*-ésima neurona de la capa anterior también se podrían expresar como:

$$\frac{\partial E}{\partial wij} = -Oi\delta j$$

donde análogamente O_i representa el valor de output de la neurona *i*-esima de la capa anterior.

En este punto se tiene la dificultad de que no se conoce el valor deseado para el output de la neurona *j* de la capa oculta, y no se puede estimar δ tan fácilmente como en el caso anterior. Se estima el mismo teniendo presente que la contribución de la *j*-ésima neurona de la capa oculta al error total del modelo depende tanto de cómo el output de esta neurona afecta al error como de la influencia de este output en la activación de las neuronas de la capa posterior. Así se estima δ [11] como:

$$\delta j = f'(aj)\sum \delta k w jk$$

donde:
δ_k = valor de delta de la neurona k-ésima de la capa posterior
w_{jk}= peso de la conexión de la neurona k-ésima de la capa superior con la neurona j.

[11] Esta ecuación justifica el nombre de retropropagación para este tipo de modelos. En cada paso para conocer el error cometido en l a capa oculta se debe conocer primero el error cometido en la capa posterior.

III. Así , para calcular los nuevos pesos en las neuronas de la capa de salida se estima el valor de Δw_{ij} como:

$$\Delta wij = -\alpha \frac{\partial E}{\partial wij} = \alpha Oi \cdot f'(aj)(yj - Oj)$$ donde α es la ratio de aprendizaje

y análogamente para los pesos de las neuronas de la capa oculta se tiene :

$$\Delta wij = -\alpha \frac{\partial E}{\partial wij} = \alpha Oi \cdot f'(aj) \sum_k \delta kwjk$$

Estas dos ecuaciones son conocidas como la ***regla de la delta*** o ***regla de Widrow-Hoff***

Finalmente, el algoritmo responsable del proceso de cálculo deberá realizar las tareas básicas:

```
inicilizar_pesos()
    repetir
        calcular_output_neuronas()
        determinar_delta_capa_de_salida()
        determinar_delta_capa_oculta()
        asignar_nuevos_pesos()
    hasta
        error_suficientemente_pequeño()
```

La velocidad de aprendizaje de la red neural viene determinado por el valor del coeficiente α. Si su valor es demasiado alto, no es posible conseguir la convergencia hacia el error mínimo; y si su valor es demasiado bajo el proceso de convergencia puede ser computacionalmente costoso y hacerse ineficiente.[Gurney,1997].. Si la topología definida por la superficie continua del error en el espacio multidimensional de los pesos presenta una forma “canalizada” en dirección al mínimo puede asumirse una alta velocidad de aprendizaje sin pérdida de estabilidad en la convergencia, pero si es “ondulada”, con presencia de mínimos locales, la velocidad de aprendizaje debe de ser menor. Así , se puede mejorar la eficiencia del proceso de cálculo introduciendo técnicas de aprendizaje adaptativo , mediante el uso del término constante *momento* en la regla de la delta ,que puede tomar valores reales entre 0 y 1. Para el cálculo del Δw en la *n*-ésima época del proceso se tiene:

$$\Delta w_{(n)} = \alpha \delta_{(n)} O_{(n)} + \lambda \cdot \Delta w_{(n-1)}$$

donde $\Delta w_{(n-1)}$ es el valor de Δw en la época anterior. Así se puede estimar el gradiente como suma recursiva de los gradientes evaluados anteriormente. Esto acelera la velocidad de aprendizaje en las topologías que permitan la rápida convergencia, pero también la disminuye cuando ésta presenta ondulaciones, dado que los diferentes signos aritméticos del valor de Δw obtenidos en las épocas anteriores tendrán tendencia a compensarse mutuamente.

4 EL PROBLEMA CLASIFICATORIO DE LOS BASALTOS.

4.1 Diferentes esquemas clasificatorios de los basaltos. Clasificaciones normativas y modales. Clasificaciones geoquímicas simples.

Se conoce como basaltos a un grupo de rocas magmáticas con bajo contenido en SiO_2 (entre el 45 – 52 %), mineralógicamente corresponden a rocas ricas en el mineral plagioclasa (en torno al 50% de la roca), con cantidades variables de otros minerales como cuarzo, biotita, ilmenita, hornblenda, olivino, clinopiroxeno, ortopiroxeno y feldespatoides (la presencia de alguno de estros minerales como el cuarzo es incompatible con la presencia de otros como los feldespatoides). En ocasiones la roca presenta una matriz de vidrio, con lo que algunos de estos minerales pueden no haber cristalizado. [Hall,1998], [Hulburt and Klein,1985]

Tradicionalmente se han usado dos tipos de criterios diferentes para la clasificación de los basaltos:

- Minerales presentes en la roca (composición mineral **modal**) : este tipo de esquema de clasificación presupone que los minerales que aparecen en la roca y las proporciones relativas entre ellos son la expresión de la composición química de la misma, pero presenta la dificultad que en ocasiones el rápido enfriamiento de un magma (por ejemplo, en el caso de una roca volcánica) produce la interrupción de los procesos de cristalización mineral que se esperaría o bien los minerales que aparecen son tan pequeños que se hacen difíciles de reconocer con exactitud si sólo se recurre a una técnica estándar de identificación como el microscopio petrográfica.

- Composición química de la roca: la composición de la roca refleja la composición del magma basáltico que la originó. Se puede utilizar en la clasificación o bien directamente los datos de análisis en forma de óxidos metálicos o bien un conjunto de reglas que intentan predecir los minerales hipotéticamente presentes y las proporciones entre ellos que se deducen a partir de los datos de composición química (en terminología petrológica composición mineral **normativa:** conjunto y proporción de minerales presentes según las reglas de composición normativa que se siguen, se expresan en porcentaje de minerales teóricos que aparecen). Estos esquemas presentan la dificultad de que se debe recurrir a análisis químicos complejos para clasificar la roca, dificultando el trabajo de identificación “de campo” o de “muestra en mano” , y que el esquema clasificatorio introduce una segmentación algo “artificial” de lo que en realidad es un continuo; en la naturaleza la composición química de la rocas varía en un continuo, y una parcelación simple de grupos de rocas a partir de datos químicos no tiene porqué reflejar necesariamente procesos petrogenéticos ni información útil si se usan criterios simples.

4.2 Ejemplos de esquemas clasificatorios: diagramas de Yoder and Tiller, diagrama de Chayes, diagramas de Strekeisen, plots K_2O+Na_2O vs SiO_2.

Ejemplos de esquemas **clasificatorios normativos** utilizados en petrología serían el tetraedro de **Yoder and Tiller** y los diagramas ternarios de **Chayes**:

1. ***El tetraedro de Yoder and Tiller**:* en esta versión del esquema se utiliza la composición normativa de seis minerales a partir de la composición química de la roca: quarzo (SiO_2), enstatita (ortopiroxeno $MgSiO_3$), forsterita (olivino $MgSiO_4$), clinopiroxeno ,albita (plagioclasa $NaAlSi_3O_4$) y nefelina (feldespatoide $NaAlSiO_4$). La proporción relativa de los seis minerales se indica, como coordenadas, como un punto en la figura. Los vértices indican la presencia del 100% del mineral y 0% de los demás. Dos de los minerales, einstatita y albita, representan composiciones intermedias en las aristas debido a la incompatibilidad de la presencia simultánea del cuarzo con forsterita y/o nefelina en situaciones de equilibrio químico.

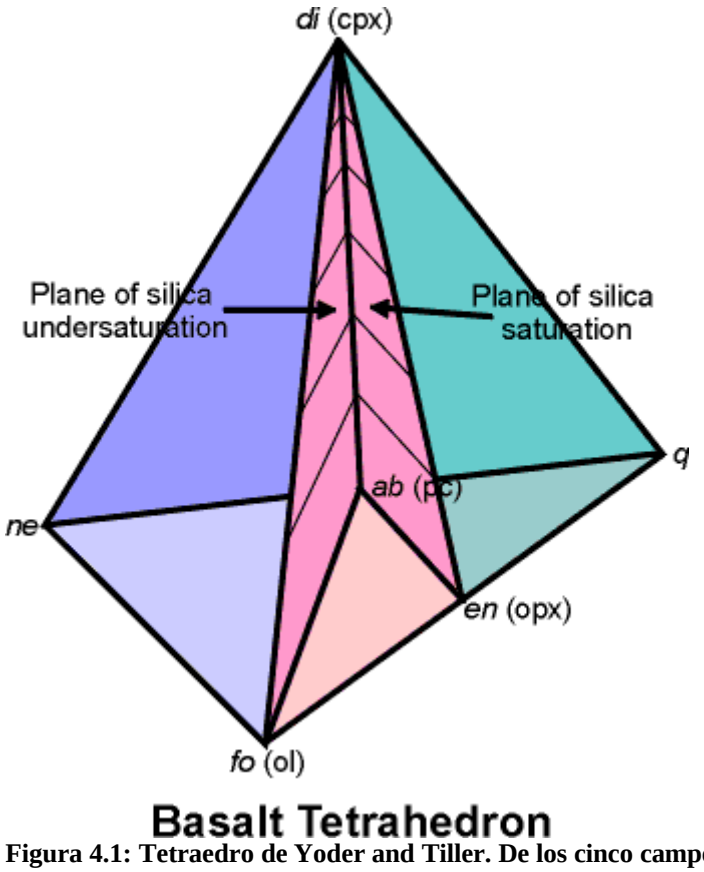

Figura 4.1: Tetraedro de Yoder and Tiller. De los cinco campos en que se divide, en la práctica se utilizan principalmente tres: el campo de toleita (verde), basalto alcalino (azul) y basalto olivínico (plano subsaturación sílice).

Según la proporción de minerales normativos, el basalto ocupa una posición en una zona del tetraedro (se puede representar como un punto dentro del tetraedro). Se distinguen cinco campos posibles:

i) Campo con cuarzo normativo (el punto se sitúa en la zona verde): el basalto se denomina **toleita**.

ii) Campo con nefelina normativa (zona azul): **basalto alcalino**

iii) Campo con forsterita+albita+clinopiroxeno (punto situado sobre el plano de subsaturación en sílice) : **basalto olivínico**.

iv) Campo albita+forsterita+clinopiroxeno+enstatita (el punto se situaría en el volumen entre los planos rosa): **toleita olivínica**

v) Campo enstatita+albita+clinopiroxeno (el punto se situaría en el planos rosa correspondiente a la saturación en sílice): **basalto hipersténico.**

En la práctica, apenas se encuentran ejemplares de los dos últimos campos. **Cabe destacar que la composición modal de la roca no tiene porqué coincidir con la composición mineralógica normativa si se sigue este esquema.**

b) ***El diagrama ternario de Chayes:*** en este esquema se usa sólo tres minerales normativos: diópsido (clinopiroxeno $CaMgSi_2O_3$), hiperestena (ortopiroxeno $FeSiO_3$) y olivino. Según la proporción relativa se distinguen dos tipos de basalto: el **basalto alcalino** (no necesariamente el mismo concepto que en tetraedro de Yoder and Tiller) y el **basalto subalcalino**. La esquinas del diagrama ternario indican la presencia del 100% del mineral y 0% de los otros dos. La proporción relativa de los tres minerales se indican por coordenadas en el diagrama ternario

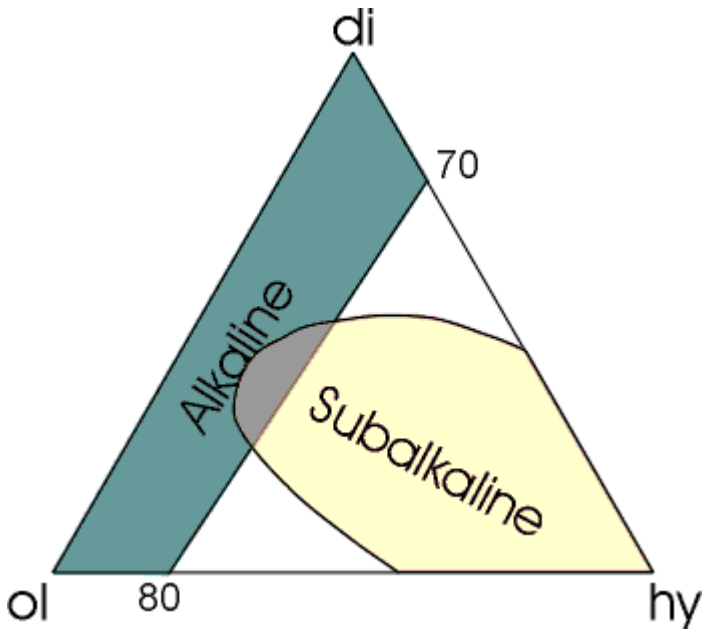

Figura 4.2: Diagrama ternario de Chayes. El campo verde corresponde a los basaltos alcalinos y el amarillo a los subalcalinos.

Por su parte, ejemplo de clasificación modal serían los diagramas ternarios y pseudoternarios de **Strekeisen**.. La IUGS ha basado su modelo de clasificación en estos diagramas :

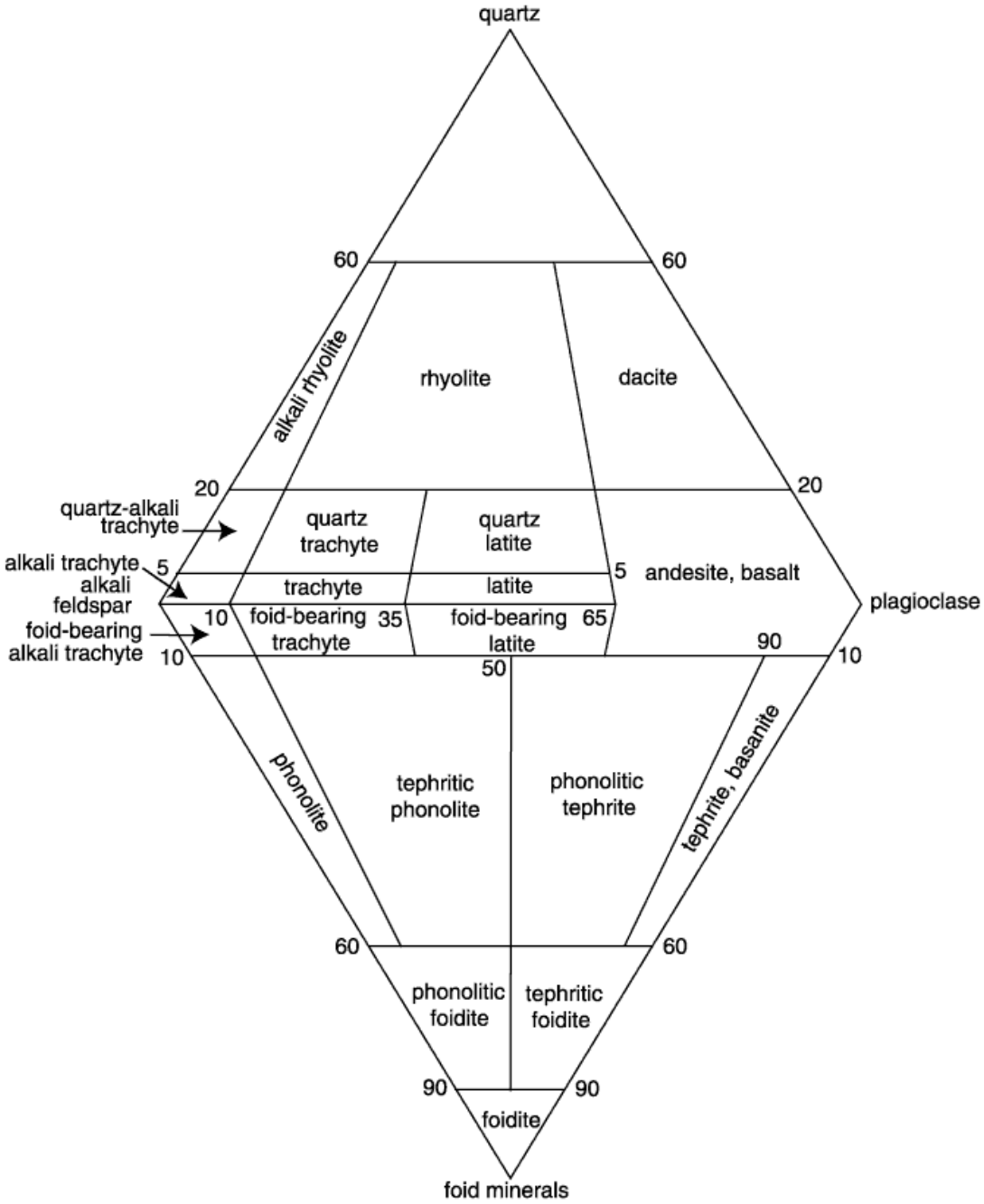

Figura 4.3: diagrama de Strekeisen para rocas volcánicas: dos diagramas ternarios superpuestos. Los extremos del diagrama superior correspondes al cuarzo, feldespato potásico en la izquierda (ortosa, sanidina o microclima) y plagioclasa en el derecho. El diagrama inferior se indican las rocas con feldespatoides, cuya presencia es simultáneamente incompatible con la del cuarzo.

Para rocas ricas en minerales máficos (hornblenda, piroxenos u olivino) se usa el siguiende diagrama pseudoternario complemenando el anterior:

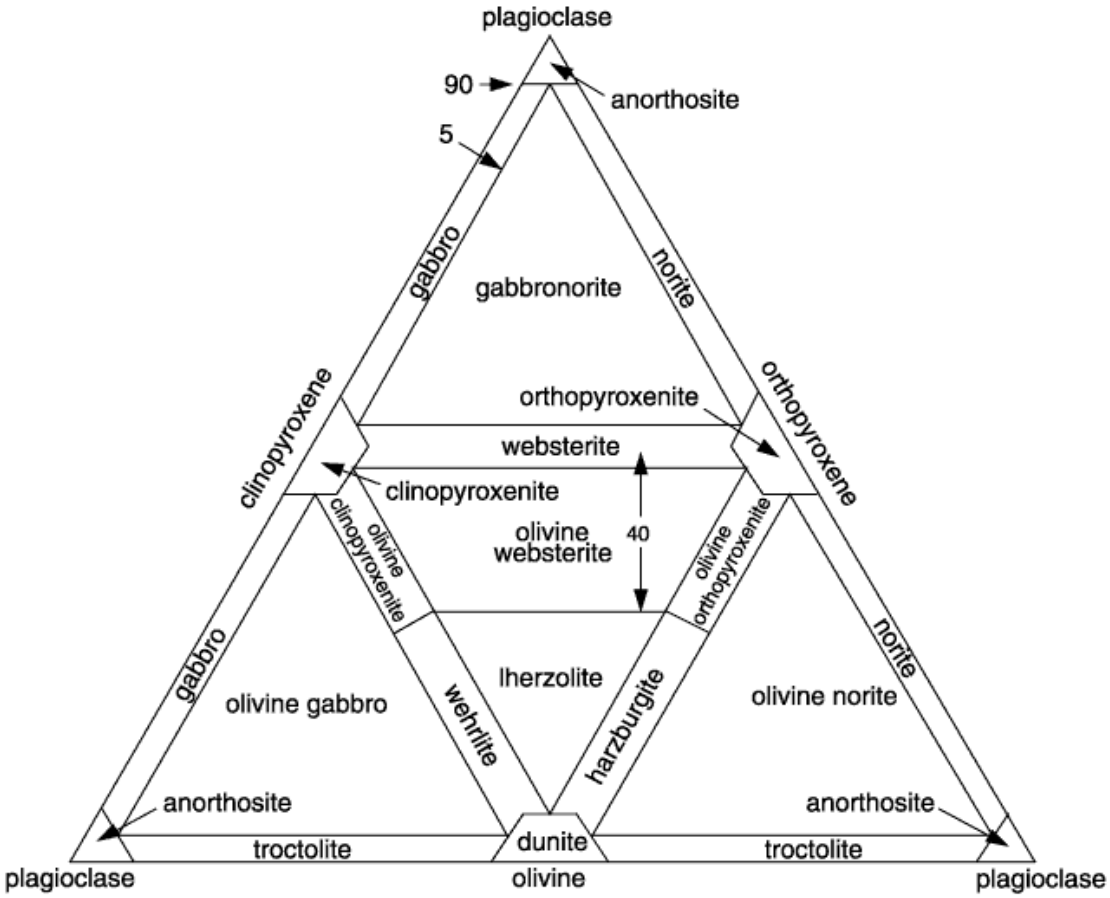

Figura 4.4: El diagrama es el resultado de la superposición de cuatro diagramas ternarios, en cuyos vértices se sitúan la plagioclasa, olivino, clinopiroxeno y ortopiroxeno (enstatita).

En el primer diagrama los basaltos quedan clasificados, aunque no diferenciados entre sí, en la derecha del mismo en las proximidades de la base triangular del diagrama. En el segundo diagrama se clasifica los gabros. El gabro es el equivalente plutónico de los basaltos, pero composicionalmente es similar. El término “gabro” se usa en petrología para referirse a rocas de composición basáltica que se han enfriado no en la superficie, como una roca volcánica o subvolcánica, sino en el interior de la corteza terrestre.

Usando exclusivamente criterios químicos puede usarse como clasificador *el **diagrama binario álcalis – sílice para rocas volcánicas***. Se hace el plot de la suma de álcalis (en forma de sus óxidos, como típicamente se dan en el análisis químico) respecto a la composición en sílice. Pos su sencillez de uso es un diagrama muy difundido. Existen varias versiones de este diagrama. Una de las más completas es la visualizada en la figura 4.5

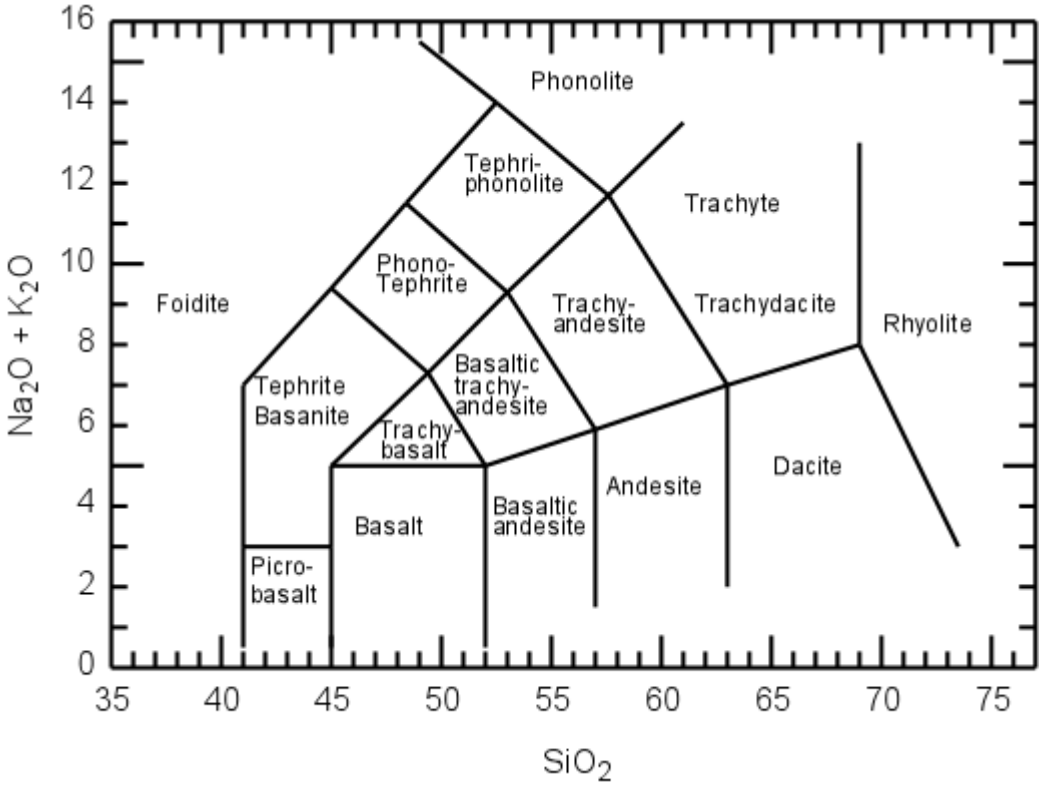

Figura 4.5: Clasificación de rocas volcánicas de acuerdo a su composición en álcalis y sílice. En este esquema los basaltos se hallan clasificados hacia el centro izquierda del diagrama: se distingue los términos basalto, picrobasalto (basalto picrítico, rico en olivino) y traquibasalto. Muy próximos se hallarían las basanitas y algunas andesitas

Esta relación de esquemas clasificatorios aquí presentada no es exhaustiva. Sólo se han citado algunos ejemplos de los esquemas clasificatorios más utilizados en la actualidad, no existiendo un único esquema universalmente aceptado y utilizado**. Una complicación adicional es que diferentes esquemas pueden usar un mismo nombre para una determinada clase de basalto, pero al ser los criterios del método de clasificación diferentes, pueden no ser necesariamente comparables en la totalidad de sus miembros**. Así, por ejemplo, la clase basalto alcalino no es exactamente la misma "clase" en el diagrama de Yoder and Tiller que en un diagrama de Chayes o en un plots de tipo *álcalis vs sílice*. Este solapamiento incompleto será uno de los factores que pueden dificultar la construcción de modelos basados en una red neural, uno de los problemas planteados por Kamath (2001).

5 DESCRIPCION DE LA BASE DE DATOS IGBA

5.1 Descripción general.

La base de datos ha sido creada por la International Union of Geological Sciences (IUGS), a cargo de la Comisión en Petrología Sistemática (CSP), con el objetivo de ser la base de datos básica en Petrología de la Rocas Ígneas. En ella se ha recopilado información publicada en revistas especializadas de petrología y geología. Se empezó a crear a partir de 1984 mediante la aportación de datos por parte de contribuyentes voluntarios, alcanzando reunir en un año más 10000 registros de análisis químicos cuantitativos de especimenes petrológicos [Chayes,1986], y en la actualidad el IGBA contiene más de 25000 descripciones de muestras rocosas, lo cual indica una ralentización en la introducción de nuevos registros, debido a la dificultad de encontrar nuevos colaboradores y al poco apoyo económico recibido, hallándose su desarrollo hoy prácticamente paralizado [Brandle, 2004, *pers.com.*]

En la actualidad la BD se compone de dos archivos: el IGBADAT5.DAT, en el que se reúnen los datos analíticos, mineralógicos y texturales de cada espécimen, y el IGBAREF5.DAT., con datos bibliográficos. El enlace relacional entre los registros de los dos archivos se hace mediante un índice de cinco dígitos. Ambos archivos, correspondientes a la versión 5ª de la base de datos, la más actualizada, pueden descargarse gratuitamente desde la web http://www.ige.csic.es/sdbp/igba.htm.

El archivo IGBDAT5 se ha estructurado como un archivo ASCII , en que cada registro lógico corresponde a un “grupo” de especimenes relacionados por su contexto geológico y geográfico, incluyéndose los datos analíticos de cada espécimen individual miembro del “grupo” .

Físicamente el archivo se estructura en una secuencia de cadenas de 80 caracteres , de longitud fija, denominadas “card” o “image” o **“card image”** en la descripción proporcionada por los creadores de la base de datos [Chayes,1986], siendo los primeros seis caracteres en todos los “card image” una clave de identificación (Id Field), mientras que los otros 74 se usan, de cuerdo con el formato para que se utiliza en cada caso, para texto libre o en formato fijo.

Cada registro lógico o “grupo” contiene varios “card images” (figura 5.1) , siendo los dos primeros la cabecera del grupo (los “record preface”), de formato fijo, los cuales contienen la información común del “grupo” de especímenes. El primero de ambos es el **RTC** o **Record Title Card**, mientras que el segundo es el **RRLC** o **Record Reference and Location Card**. A continuación aparecen secuencialmente los “card image” que describen a cada espécimen individualmente . Para describir cada espécimen individual se debe usar tres o más “card images” .

Los primeros dos “card image” de un espécimen individual tienen formato fijo y son de aparición obligatoria, y contienen información sobre:

- Información general del espécimen y localización geográfica del mismo. (corresponde secuencialmente al primer *card image* , el tipo ‘A’)
- Composición química, expresada en forma de óxidos, de los componentes principales (en terminología petrológica :”elementos mayoritarios”) del análisis de la roca. (corresponde al siguiente *card image* o *card image* ‘B’)

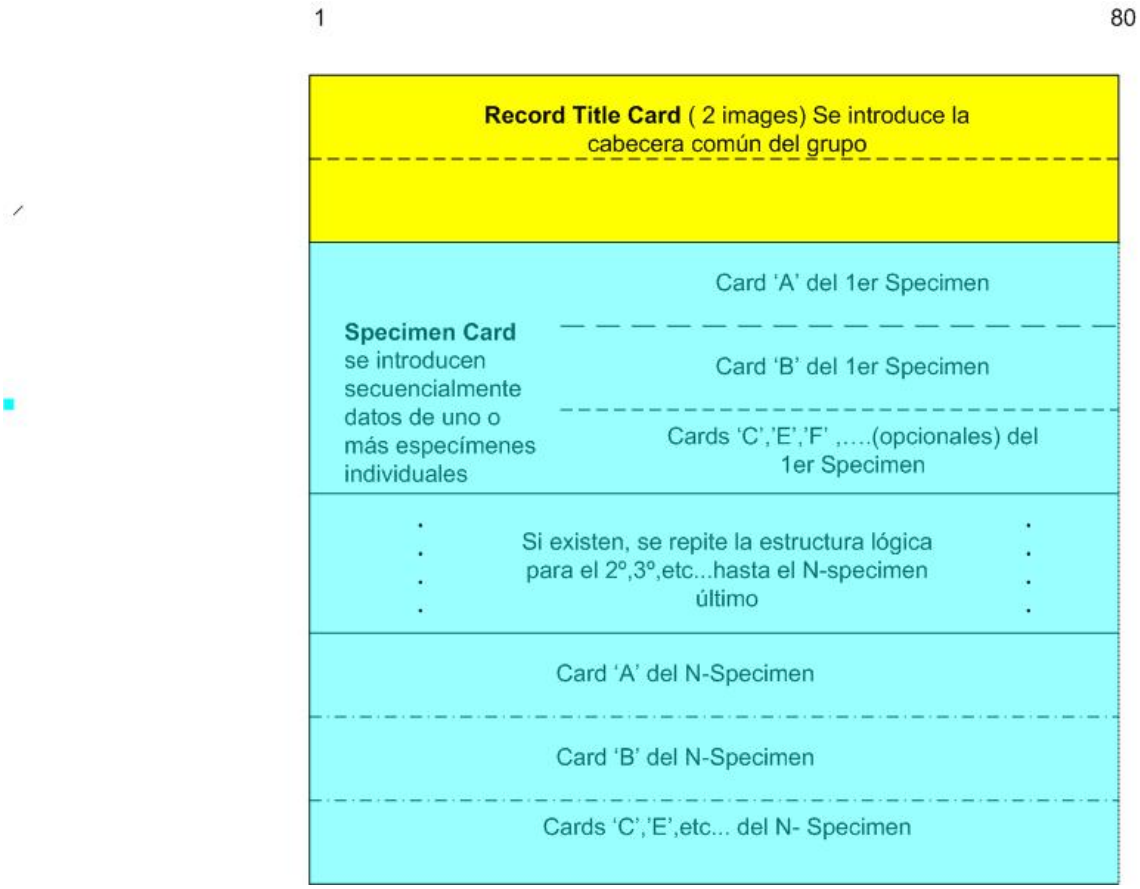

Figura 5.1: Estructura de un registro lógico en la base de datos. Corresponde a una secuencia de cadenas de longitud fija de 80 caracteres ASCII; los caracteres de la cadena se representa horizontalmente y la secuencia de cadenas verticalmente. Las dos primeras cadenas, en amarillo, corresponde a la cabecera del grupo, y son continuadas por una secuencia de cadenas en que se describen las características de cada espécimen individual.. Al menos en un grupo debe existir un espécimen, con una cadena tipo ‘A’ seguida de otra tipo ‘B’. El resto de cadenas son opcionales y sólo aparecen cuando hay información disponible. en la descripción del espécimen.

Mientras que, para cada espécimen individual, los *card image* del tercero en adelante sólo son de aparición opcional y en ellos se incluye ,siguiendo un orden secuencial, la información disponible sobre:

- Status ,códigos que enlazan con información predeterminada en un anexo de la base de datos..
- Elementos traza y minoritarios analizados.
- Periodo geológico en que la muestra a sido datada
- Minerales que la constituyen
- Información adicional

corresponden a los *card images* con símbolos de identificación alfabéticos según el orden: 'C','D',....,'Z'.

El campo de identificación de todos los *card image* (Id Field, los primeros seis caracteres de cada "card image") se subdividen en tres partes:

- Caracteres 1-3: campo del Identificador del Registro ("Record Identifier" o RS), constituido por al menos un carácter alfabético y un máximo de tres caracteres alfabéticos justificados a la derecha del campo. Tiene el mismo valor en todos los "*card image*" de un mismo registro lógico.

- Caracteres 4-5: se dejan blanco en los dos *card images* iniciales de un grupo, (correspondientes al *card prefac*e),, mientras que toman un valor alfanumérico (IS) en los *card images* que describen especimenes individuales, que permite distinguirlos entre sí al tomar valores diferentes para cada espécimen del grupo.

- Carácter 6: su valor identifica el tipo de *card image:* al primer *record preface* o RTC le corresponde el carácter '1', al segundo *record preface* o RRLC el carácter '2', mientras que en los *card images* correspondientes a especimenes individuales dentro del grupo le corresponden letras mayúsculas: una 'A' al primer *card image* con la descripción general, una 'B' para el *card image* que guarda la información del análisis químico de componentes mayoritarios, y si existen más *card images* que describen el espécimen se continua la secuencia con 'C' ,'D',...'Z',.

En el presente proyecto serán utilizados sólo los *card image* tipo 'A' y 'B' de los especimenes individuales , por lo que no se describe con detalle la estructura del resto.[12]

[12] Su descripción puede encontrarse en Brandle and Nagy (1995)

5.2 Descripción detallada de los campos del primer card image del record preface de grupo: Record Title Card (RTC)

Contiene cuatro campos de longitud fija: campo RS, campo en blanco, campo '1' y campo TITL (figura 5.2 y tabla 5.1):

Figura 5.2: estructura lógica de los campos del RTC y sus posiciones en la cadena de caracteres.

CAMPO	Posiciones que ocupa	INFORMACION QUE CONTIENE
RS	1-3	clave Id del grupo dentro del archivo. Corresponde a tres caracteres alfabéticos en mayúsculas
	4-5	no se usa
'1'	6	contiene el carácter ASCII '1'. Identifica el tipo de *card image*: en este caso la primera del *record preface*
TITL	7 -80	campo que contiene, en formato libre, una breve descripción del grupo: puede corresponder a la descripción su situación geológica, título del trabajo de donde se ha extraído, una breve referencia petrológica o geográfica,...etc

Tabla 5.1. Significado de los diferentes campos del RTC

5.3 Descripción detallada de los campos del segundo card image del record preface de grupo: Record Referente and Location Card (RRLC)

Contiene diez campos en total, de longitud fija,, de los cuales dos no se usan y resume la información del contexto geográfico y geológico del grupo de muestras, así como el código de enlace con el archivo IGBAREF. (figura 5.3 y tabla 5.2)

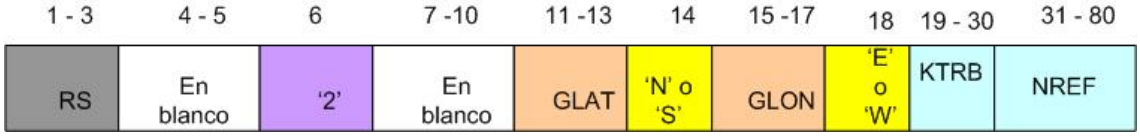

Figura 5.2. Estructura lógica de los campos de la cadena de caracteres RRLC y su posición en la misma

CAMPO	Posiciones que ocupa	INFORMACION QUE CONTIENE
RS	1-3	clave Id del grupo dentro del archivo. Corresponde a tres caracteres alfabéticos en mayúsculas (igual al RS del RTC)
	4-5	no se usa
'2'	6	contiene el carácter ASCII '2'. Identifica el tipo de *card image*: en este caso la segunda del record *preface*
GLAT	11 – 13	latitud, en grados, del espécimen del grupo situado más al norte
'N' o 'S'	14	indica si la latitud es Norte o Sur (respectivamente, carácter 'N' o 'S', no toma ningún otro valor)
GLON	15-17	longitud en grados, la más cercana al Este del grupo
'E' o 'W'	18	indica si los grados de longitud geográfica son Este o Oeste
KTRB	19 - 30	nombre del autor
NREF	31 – 80	códigos de mínimo una hasta 10 diferentes referencia bibliográficas, corresponde a la relación con el Id de los registros de publicaciones del archivo IGBREF

Tabla 5.2. Significado de los campos en la línea RRCL

5.4 Descripción detallada del card image 'A' de cada espécimen .

Esta línea es la primera que describe a un espécimen individual dentro del grupo. Se indica en este campo la localización geográfica detallada del espécimen analizado, así como su clasificación petrológica de la muestra según el autor del que se ha tomado la referencia. (figura 5.4 y tabla 5.3):

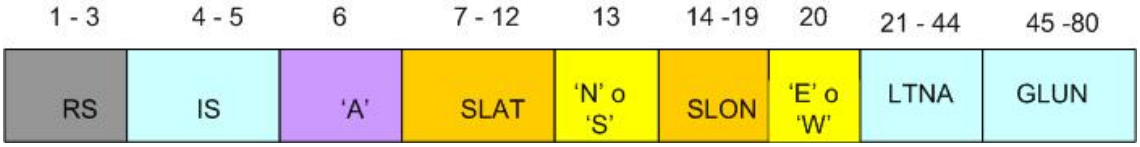

Figura 5.4. Estructura lógica de los campos de formato fijo en la cadena de caracteres tipo card-'A' y su posición en la misma

CAMPO	Posiciones que ocupa	INFORMACION QUE CONTIENE
RS	1-3	clave Id del grupo dentro del archivo. Corresponde a tres caracteres alfabéticos en mayúsculas
IS	4 – 5	clave Id del espécimen dentro del grupo, corresponde a dos caracteres alfabéticos diferentes para cada ejemplar del grupo, pero que no tienen porqué estar ordenado alfabéticamente entre especimenes de un mismo grupo necesariamente .
'A'	6	contiene el carácter ASCII 'A', indicativo de que este *card image* describe el nombre del espécimen y su localización
SLAT	7 -12	x100, indica en milésimas grados la latitud en que se localizó la muestra. Las milésimas de grado están justificadas a derecha. Si no se conoce con precisión de milésimas de grado, se dejan en blanco el carácter a justificar a derecha y se escriben sólo los valores conocidos
SLA	13	indica con 'N' o 'S' si la latitud es Norte o Sur
SLON	14 – 19	x1000, indica en milésimas de grados la longitud en que se localizó la muestra. Las milésimas de grado están justificadas a derecha. Si no se conoce con precisión de milésimas de grado, se dejan en blanco el carácter a justificar a derecha y se escriben sólo los valores con precisión conocida
SLO	20	indica con 'E' o 'W' si la longitud geográfica es Este u Oeste
LTNA	21-44	nombre que da el autor a la roca, literalmente tal como aparece en su publicación
GLUN	45 - 80	nombre de la unidad geológica en la que se ha recogido el espécimen, tal como lo da el autor

Tabla 5.3. Descripción de los campos de la línea 'A'

5.5 Descripción detallada del card image 'B'.

Contiene la información del análisis químico de los componentes mayoritarios del espécimen, en forma de óxidos, con precisión de hasta el 0.01%, así como la clave de clasificación de la roca (figura 5.5 y tabla 5.4), de acuerdo al esquema de nombres de la IUGS en el campo RKNUM (ver Anexo I para lista de códigos) :

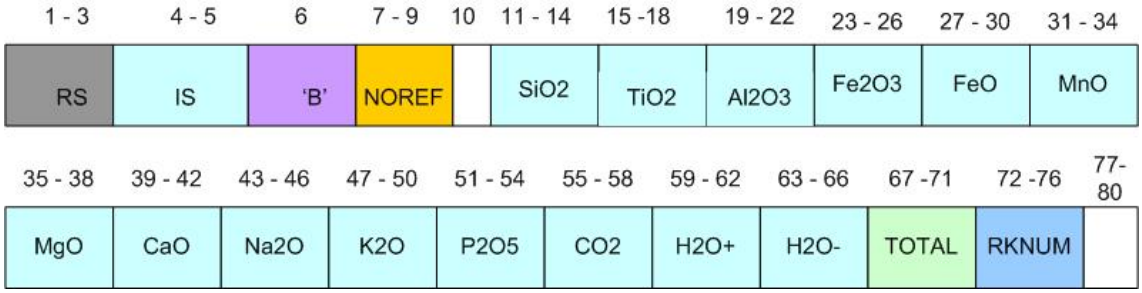

Figura 5.5. Estructura lógica de los campos de formato fijo en la cadena de caracteres tipo card-'B' y su posición en la misma. El último campo (caracteres 77 hasta 80) de la cadena, no se usa.

CAMPO	Posiciones que ocupa	INFORMACION QUE CONTIENE
RS	1-3	clave Id del grupo dentro del archivo. Corresponde a tres caracteres alfabéticos en mayúsculas
IS	4 – 5	clave Id del espécimen dentro del grupo, corresponde a dos caracteres alfabéticos diferentes para cada ejemplar del grupo, pero que no tienen porqué estar ordenado alfabéticamente .
'B'	6	contiene el carácter ASCII 'B', indicativo de que este *card image* describe los componentes principales análisis químico de la roca en forma de óxidos.
NOREF	7 - 9	orden de la muestra dentro de la secuencia NREF de la *card image* 'A'
	10	carácter espacio blanco, no se usa
SIO2	11 - 14	contenido de SiO_2, en centésimas de porcentaje de peso. Se justifica el valor proporcionado por el autor en la derecha del campo. Si se da el valor con precisión menor de 0.01%, se dejan los espacios en blanco necesarios (Un valor de 52.10 % de SiO_2 se entraría como el valor= 5210, pero el valor 52.1% se entra como valor=521[blanco], mientras que el valor 5.21% se entraria como valor=[blanco]521
TIO2	15 - 18	análogamente para el TiO_2

Al2O3	19 - 22	análogamente datos del Al_2O_3
FE2O3	23 - 26	datos para el Fe_2O_3, que representa la presencia del catión Fe^{3+}
FEO	27 - 30	si se dispone de datos diferenciados, se entra el valor de FeO para indicar el contenido de Fe^{2+}. Con frecuencia, no se diferencian en los análisis, y todo el hierro se entra como Fe_2O_3
MNO	31 - 34	contenido en MnO, para indicar la composición de manganeso
MGO	35 - 38	contenido en MgO
CAO	39 - 42	contenido en CaO
NA2O	43 - 46	contenido en Na_2O
K2O	47 - 50	contenido en K_2O
P2O5	51 - 54	contenido en P_2O_5
CO2	55 - 58	contenido en CO_2, típicamente representa la presencia de carbonatos
H2O+	59 - 62	presencia de agua, ocupando posiciones cristaloquímicas en los minerales de la roca.
H2O-	63 - 66	presencia de agua, adsorbida en la roca. Se elimina por debajo de 110ºC
TOTAL	67 - 71	suma del porcentaje total dado por el autor. Suele ser próximo al 100%, pero raramente coincide exactamente cuando se expresa el análisis en forma de óxido
RKNUM	72 - 76	número correspondiente al nombre dado a la roca, según el sistema de códigos proporcionado por el IUGS (ver anexo I)

Tabla 5.4. Descripción de los campos de la línea 'B' de cada espécimen

5.6 Campos de los image cards *"C", "D" , "E" y sucesivos.*

Tienen estructura variable de acuerdo a la información que contengan. Al no ser utilizados en el presente trabajo no los describo en detalle. Se puede encontrar también su descripción completa en el documento del IGBA:

ftp://ftp.csic.es/pub/igneous/structur.txt.

Contiene secuencialmente seis campos diferentes , de formato variable. La separación entre campos se realiza utilizando como carácter separador el símbolo ASCII **':'** a la derecha del campo, de modo que el contenido de cada campo, salvo el primero, se halla entre dos caracteres **':'** . Cuando en un campo hay un listado, se usa el carácter *coma* ',' como separador entre los elementos de la lista. Los campos pueden estar vacíos.

El contenido de los mismos, secuencialmente es:

- Campo **“status”:** lista de códigos de dos caracteres. Estos códigos codifican información diversa sobre la muestra y los métodos analíticos utilizados de acuerdo a un sistema de códigos. Los valores de código posibles están descrito en la lista de códigos preestablecida para este campo.

- Campo **“elementos traza”:** lista de análisis de elementos minoritarios no contemplados en la lista del *card image* ‘B’ y elementos traza , si se han llevado a cabo (se indican con su símbolo químico de la Tabla Periódica de los Elementos, en mayúsculas). Se debe indicar el tipo de unidad de concentración química que se está utilizando.

- Campo **“edad ”.** Contiene dos o más subcampos separados por el carácter ASCII **‘;’** El primer subcampo, que puede estar vacío, corresponde a la edad de la roca referida al sistema de unidades cronoestratigráficas, de acuerdo a una lista de códigos. El segundo y posteriores corresponde a los subcampos “edad física”, que indican dataciones de la edad absoluta de la roca, en cifras, cuando ésta está disponible y también el método de datación utilizado , de acuerdo a un código preestablecido para el mismo.

- Campo **“descriptores petrográficos”:** lista de códigos de dos letras que codifican información textural de la roca y sobre su estado de alteración química, de acuerdo a la lista de códigos preestablecida.

- Campo **“asociación mineral”:** lista de códigos que codifica los minerales presentes en la roca, con información textural sobre los mismos, de acuerdo a la lista de códigos preestablecida.

- Campo **“información adicional” :** contiene ,en formato libre de hasta un total de 500 caracteres, bloques de información con cualquier información adicional que se considere relevante . Los bloques se hallan delimitados entre dos parejas de caracteres de paréntesis. El objetivo de este campo es incluir toda la información que se considera relevante y no ha podido ser incluida en ninguno de los otros campos de los registros de la base de datos.

A continuación , en la tabla 5.5 , se describe detalladamente como ejemplo un registro lógico del archivo IGBADAT tal como aparece en el mismo. Corresponde a un registro de un grupo de rocas granodioríticas situadas en la provincia de Jiang (China),con el código de grupo CCH en el cual hay sólo dos especimenes incluidos en el grupo, descritas con los códigos CCH A y CCH B:

EJEMPLO DE REGISTRO LÓGICO EN IGBDAT5

```
CCH  1PORPHYRY COPPER DEPOSIT IN HEILONGJIANG,CHINA                              (1)
CCH  2     28N118EXU JIANGUO   4354                                              (2)
CCH AA 28000N118000EGRANODIORITE            HEILONGJIANG,CHINA                   (3)
CCH AB  1 6178  361642 223 343  08 269 44  443 206  2              9808 1490     (4)
CCH AC1D,2E,4D,4J,4K::;292E6-KAR/PB:BU,EY,IA,IM,IZ,JB:NA:((XL DUBAOSHAN PORPHYRY (5)
CCH AD COPPER DEPORSIT,HEILONGJIANG PROVINCE))((XP CH08)):                       (6)
CCH BA 28000N118000EGRANITE-PLAGIOCLASE     HEILONGJIANG,CHINA                   (7)
CCH BB  1 7078  161592  79  96  08  59 223 528 228  1              9917   20     (8)
CCH BC1D,2E,4D,4J,4K::;245E6-KAR/PB:BU,EY,IA,IM,IZ,JB:NA:((XL DUBAOSHAN PORPHYRY (9)
CCH BD COPPER DEPORSIT,HEILONGJIANG PROVINCE))((XP CH08)):                       (10)
```

(1) = Línea RTC o ‘1’ de la cabecera grupo con índice id=CCH. Para un grupo de rocas porfídicas situadas en China

(2)= Línea RRLC o ‘2’ de la cabecera. Indica la localización geográfica del grupo y una referencia del archivo IGBAREF

(3)= Línea ‘A’ del primer espécimen del grupo (espécimen con el 5º carácter id=A en la línea), con su localización geográfica exacta.

(4) =Línea ‘B’ del primer espécimen del grupo. Contiene su análisis químico (faltan los tres últimos óxidos) . El tipo de roca se especifica como RKNUM=1490 (una granodiorita,ver Anexo I)

(5) y (6) = Linea ‘C’ y ‘D’ con información adicional. Secuencialmente y separados los campos por el símbolo ‘:’ aparecen : el campo “status”, no hay campo “elementos traza” y el campo “edad” el subcampo “edad cronoestratigráfica” está vacio, pero sí aparece el subcampo “edad física” con una datación y sus métodos de determinación, el campo “descripción petrográfica” con su lista de descriptores separados por comas, el campo “asociación mineral” (indica sólo el símbolo NA= no hay información)
y finalmente el campo “información adicional” con dos bloques.

(7) = Linea ‘A’ del segundo espécimen del grupo (espécimen con el 5º carácter id=B en la linea), con su localización geográfica exacta.

(8) = Línea ‘B’ del primer espécimen del grupo. Contiene su análisis químico (faltan también los tres últimos óxidos) . El tipo de roca se especifica como RKNUM=20 (sin nombre en el sistema IGBA)

(9) y (10) = Líneas ‘C’ y ‘D’del segundo espécimen. Con estructura similar al primer espécimen.

Los códigos completos de los campos están disponibles en el documento previamente mencionado: ftp://ftp.csic.es/pub/igneous/structur.txt.

Tabla 5.5. Ejemplo de registro lógico en IGBADAT y descripción del mismo (en verde).

6 ANÁLISIS DEL SOFTWARE UTILIZADO EN EL ESTUDIO: EL PAQUETE DE PROGRAMAS WEKA.

WEKA[13] es una aplicación desarrollada por la Universidad de Waikato (Nueva Zelanda), escrita en lenguaje Java para plataformas Linux, Windows y Macintosh. [Witten & Frank,2000]. Puede descargarse libremente en la web http://www.cs.waikato.ac.nz/~ml/weka/index.html. Incluye toda una serie de funcionalidades:

- Implementación de amplia variedad de algoritmos de *machine learning*, entre los cuales está disponible el algoritmo clasificatorio utilizando redes neurales de retropropagación, con varios parámetros modificables por el usuario.

- Inclusión de una amplia variedad de herramientas de preproceso de los conjuntos de datos, (denominadas *filtros* en la documentación del programa). Podría llegar a ser posible preprocesar los datos, construir un modelo con ellos y analizar la eficiencia del modelo sin necesidad de crear ninguna aplicación adicional de ayuda.

- Existe disponible una amplia documentación on-line, que va siendo actualizada periódicamente a medida que se introducen cambios y se amplia la aplicación. Entre la documentación disponible existe documentación API generada mediante Javadoc, la cual describe exhaustivamente las clases Java que se usan en el proyecto. Esto permite que el usuario avanzado pueda llegar a desarrollar e implementar en la aplicación sus propios algoritmos y filtros de preprocesado de datos. Así , por ejemplo, para implementar un algoritmo clasificador puede usarse las clases del paquete java *classifiers. L*a clase diferida *Classifier* define la estructura que debe tener cualquier esquema clasificatorio que se quiera implementar, como subclase. Para ello debe implementarse el método abstracto *buildClassifier()* en la subclase.

- La aplicación puede usarse tanto desde la línea de comandos del sistema operativo como desde las interficies gráficas que proporciona.

- Dispone de herramientas de visualización y análisis gráfico de los modelos construidos, lo cual facilita la toma de decisiones respecto la validez de los mismos

La implementación de las redes neurales se lleva a cabo en el paquete **weka.classifiers.functions.neural.** En este paquete se halla implementada la clase ***weka.classifiers.functions.neural.NeuralNetwork*** como subclase de *Classifiers*.

[13] Acrónimo de *Waikato Environment for Knowledge Analisys*

Algunos de los métodos[14] relacionados con las opciones de cálculo y entrenamiento de la red que incorpora esta clase son:

- La normalización de atributos numéricos (entre –1.0 y +1.0). Es indicado por el método *setNormalizeAtributes()*. Por defecto, los atributos numéricos son siempre normalizados.

- Definición del número de épocas que se usarán en el entrenamiento: *setTrainingTime()* (Por defecto ,500)

- Establecimiento de la ratio de aprendizaje α de los nodos de la red: *setLearningRate()* (entre 0 y 1, por defecto se usaría 0.3)

- Permite establecer un momento λ de aprendizaje adaptativo: mediante el método *setMomentum().* (Por defecto se usa 0.2).

- Inclusión de un método de decaimiento *(decay*) mediante el cual disminuye la velocidad de aprendizaje a medida que progresa el cálculo del modelo, al dividirse el coeficiente inicial *ratio de aprendizaje* α por el número de épocas y actualizarse con este nuevo valor al empezar una nueva época. No se usará en el presente trabajo.

- Determina el tamaño porcentual del conjunto de validación: *setValidationSetSize()*. Por defecto, no se usa un conjunto de validación.

- *setRandomSeed(),* generador de números aleatorios. Se usa para determinar los pesos iniciales en cada nodo

Por su parte, las clases **SigmoidUnit** y **LinearUnit** implementan la interficie ***NeuralMethod*** y se usan para calcular el output de la función de activación de las neuronas, usando respectivamente o bien una función sigmoidal o una función lineal, así como también para calcular Δw y los nuevos pesos de las neuronas en el modelo. En la figura 6.1 se observa la interfície gráfica que proporciona el paquete para la construcción de modelos clasificatorios mediante redes neurales de retropropagación. En la imagen se puede observar dos ventanas desplegadas correspondientes una a la selección del clasificador , los atributos de análisis y métodología del entrenamiento y la otra es un editor de los parámetros que se usarán en el entrenamiento de la red neural.

[14] Término “método” usado en el contexto de metodología de programación orientada a objetos, procedimiento o función ligado a una clase.

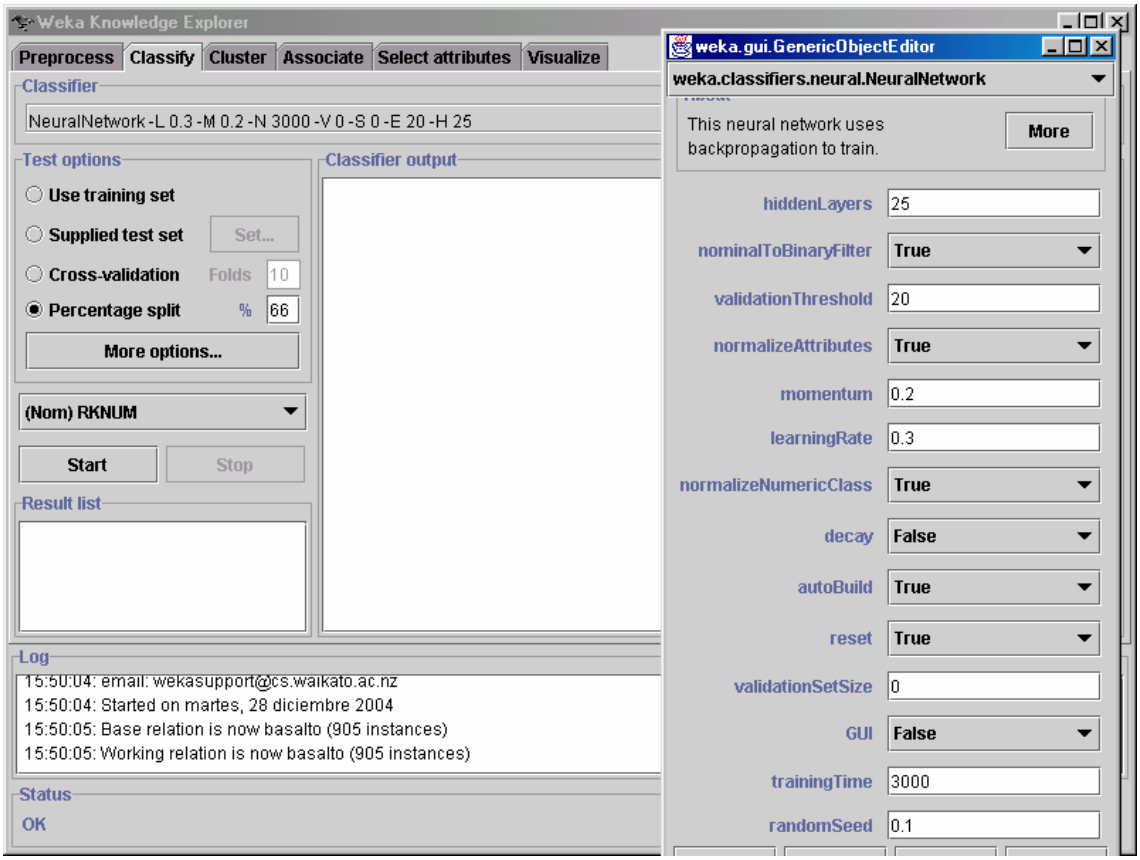

Figura 6.1. Interficie gráfica de WEKA en la cual el usuario puede introducir y/o seleccionar los parámetros del análisis. Se ha seleccionado el clasificador NeuralNetwork en la imagen. En la ventana de la derecha se introducen los parámetros del mismo.

En el menú de los clasificadores, el clasificador *NeuralNetwork*. En el menú situado a la izquierda puede seleccionarse también que método se elegirá para construir el **conjunto de evaluación de** entre los datos que se usarán para el análisis; se proporcionan cuatro métodos:

- **"use training set":** se efectúa la evaluación del modelo usando los mismos que se usan para entrenar la red.
- **"suplied test set":** el usuario indica específicamente que conjunto de datos debe de ser utilizado como conjunto de validación.
- **"cross validation":** la aplicación usa la técnica de evaluación cruzada para evaluar el modelo.
- **"percentage split":** se indica el porcentaje de datos sobre los que se efectúa el entrenamiento de la red, dejándose el resto como conjunto de evaluación. Éste será el método usado en el presente trabajo.

Por debajo de este menú se puede seleccionar cual es el atributo objetivo en la construcción del modelo clasificatorio . (No desplegado en la imagen)

En el menú desplegado a la derecha se pueden introducir el resto de parámetros específicos que se usarán en el análisis. Entre las opciones disponibles en el menú se tiene:

- Número de capas ocultas en la red
- Indicación de que deben usarse o no atributos normalizados, tanto para atributos representando tipos de datos categóricos como numéricos.

- Valor del coeficiente α ratio de aprendizaje y del momento λ que se usará.
- Se indica si se quiere usar un **conjunto de validación.**
- Se indica si se corrige α con un decaimiento a medida que avanza el entrenamiento de la red.
- Opción GUI: se usa una pantalla gráfica para diseñar la arquitectura de la red. Sin embargo, uso puede presentar problemas de errores de "cuelgue" del programa al realizar análisis sucesivos[15] en una misma sesión de trabajo, por lo que no se usa esta opción en las pruebas.
- Se indica el número máximo de épocas que se usarán en el entrenamiento de la red.
- Se proporciona una "semilla" inicial al método Java generador de números aleatorios.

Los datos que se usarán en el análisis deben introducirse mediante un archivo de texto ASCII en formato ARFF. En este formato de archivo, las instancias de datos, correspondientes a registros de la base de datos son independientes entre sí, no implicando el orden del listado de las mismas ningún orden ni relación entre ellas necesariamente [Witenn, Frank,2000]. El formato ARFF permite dos tipos de datos básicos: el tipo NOMINAL (datos categóricos) y el tipo NUMERIC (datos numéricos). La aplicación los puede interpretar de modo distinto según la técnica de *machine learning* que se esté utilizando y el modelo que se esté construyendo . La estructura del mismo es simple consistente en un listado de cadenas de caracteres, y puede dividirse en tres partes:

- **Cabecera**. Indicada por una única cadena inicial de tipo:

  @relation nombre_de_la_relacion

- **Lista secuencial ordenada de atributos** . Se indica por la lista de cadenas tipo:

  @attribute nombre_de_atributo tipo_de_atributo

  Cada cadena de la lista indica las característica y nombre de cada uno de los atributos del archivo.

- **Datos.** Se introduce indicando la cadena:

  @data

Debajo de esta línea aparecen listadas secuencialmente las instancias. Cada línea del archivo de texto corresponde a una instancia de datos. El valor de los atributos se indica en la línea de cada instancia de acuerdo al mismo orden determinado por la lista de atributos. Los valores se separan entre sí por un carácter ASCII coma **','** . Pueden introducirse comentarios en el archivo, la línea de comentario debe de ir precedida por el carácter ASCII '**%**'[16].

[15] Sin duda, un *bug* de esta versión de WEKA.
[16] Puede verse la cabecera del archivo ARFF del proyecto en el capítulo 7.

7 TRATAMIENTO DE DATOS.

El tratamiento de los datos de la base IGBADAT, para la obtención de un archivo con formato ARFF, que incluya los registros y los atributos necesarios para la construcción del modelo clasificatorio con WEKA se ha llevado a término en dos etapas:

En primer lugar ha sido necesario comprobar que el archivo IGBADAT5.DAT presenta el formato descrito por sus autores. Se han localizado algunos errores en las longitudes de las líneas de texto del archivo, supuestamente de formato fijo de 80 caracteres. En un pequeño porcentaje , hay líneas de texto con un menor número de caracteres respecto a lo esperado y en menor medida líneas de texto con algunos caracteres adicionales. Se ha podido comprobar que el error no es crítico, ya que el exceso de caracteres corresponde a caracteres "espacio en blanco" (ASCII 032) añadidos a final de línea y las líneas con menos de 80 caracteres corresponde a líneas que deberían tener espacios en blanco a final de la misma para completar los 80 del formato, faltando éstos. En ningún caso afecta el error al formato de los datos de análisis químico. Para facilitar la detección sistemática de este error se ha desarrollado una pequeña aplicación DOS denominada **prBD.exe**. El código fuente de esta aplicación se ha desarrollado en C++[17] y se ha compilado mediante el compilador **Visual C++ 6.0 de Microsoft**[18]. A continuación se lista el código fuente utilizado:

listado del código fuente de prBD.cpp

```
//////////////////////////////////////////////////////////////////////////
// pre: existe el archivo IGBA5 original, se  lee con el nombre
//      "c:\\tfc\\igbadat5BIS.dat"
// post: se crea un archivo de texto con el nombre "c:\\tfc\\igbaRes1.txt" en el cual se lista :
//      número de línea del archivo igbadat5BIS.dat con error (diferente de 80 caracteres)
//      número de caracteres reales en dicha línea
//////////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "prBD.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif


CWinApp theApp;

using namespace std;

// función principal
int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
int nRetCode = 0;
CStdioFile fp1; // apuntará a igbadat5bis.dat
CStdioFile fp2; // apuntará a igbaRes1.txt
CFileException fileException;
```

[17] Una buena de este lenguaje de programación puede encontrarse [Ceballos, 2003]
[18] Las particularidades de este compilador y sus librerías pueden encontrarse en [D`Andrea,2000]

```
// initialize MFC and print and error on failure
if (!AfxWinInit(::GetModuleHandle(NULL), NULL, ::GetCommandLine(), 0))
{
// error de inicializacion
cerr << _T("Fatal Error: MFC initialization failed") << endl;
nRetCode = 1;
}
else
{
// TODO: code your application's behavior here.
if ( !fp1.Open( "c:\\tfc\\igbadat5BIS.dat", CFile::modeRead, &fileException ) )
{
	TRACE( "Can't open file %s, error = %u\n","c:\\tfc\\igbadat5BIS.dat",
	fileException.m_cause );
	nRetCode=1;
	return nRetCode; // no existe igbadat5Bis.dat
}
if ( !fp2.Open( "c:\\tfc\\igbaRes1.txt", CFile::modeCreate+CFile::modeWrite, &fileException ) )
{
	TRACE( "Can't open file %s, error = %u\n","c:\\tfc\\igbaRes1.txt", fileException.m_cause
	);
	fp1.Close();
	nRetCode=1;
	return nRetCode; // no se puede crear igbaRes1.txt
}

CString s;
int nLinea=0; // indica la línea actual que se está leyendo
int longitudLinea=0;
do
{
	 nLinea++;
	 fp1.ReadString(s); // lee la linea actual en igbadat5bis.dat
	 longitudLinea=s.GetLength(); //obtiene la longitud de la línea leída
	if(s.GetLength()!=80)
	{
	//hay error
	CString os;
	os.Format("Error en línea %d= %d caracteres\n",nLinea,longitudLinea);
	fp2.WriteString((LPCTSTR)os);
	cout<<(LPCTSTR)os; //salida también por pantalla
	}
}while(longitudLinea!=0); //encontrado fin de archivo igbadat5bis.dat
fp1.Close();
fp2.Close();
// se cierran los archivos abiertos
	  return nRetCode; //si el analisis se ha efectuado correctamente , devuelve 0
}
```

Para facilitar la lectura de los datos analíticos, una vez localizadas las líneas con error, éste se ha corregido manualmente mediante el uso del editor ASCII TEXPAD, que permite reconocer fácilmente la longitud real de una línea de texto ASCII, suprimiéndose o añadiendo caracteres “espacio en blanco” allí donde era necesario.

En segundo lugar deben seleccionarse los registros lógicos y los atributos que formaran parte del archivo ARFF.

Los registros que se seleccionan corresponden sólo a los basaltos que pueden seguir la nomenclatura de las clases principales de Yoder and Tiller, según el valor dado a la variable RKNUM, es decir, de acuerdo al código IUGS:(ver Anexo I)

- **Basalto alcalino** , para RKNUM = 400
- **Toleita o basalto toleitico**, para RKNUM = 570
- **Basalto olivínico** , para RKNUM = 530

Los atributos seleccionados corresponden a los valores de los componentes químicos mayoritarios utilizados, así como el valor de RKNUM. Con los registros y atributos seleccionados se construye el archivo **igbaRes.arff**. La cabecera del mismo, que detalla los atributos seleccionados y el orden de aparición de los mismos en **igbaRes.arff** es la siguiente:

@relation basalto
@ATTRIBUTE SIO2 REAL
@ATTRIBUTE TIO2 REAL
@ATTRIBUTE AL2O3 REAL
@ATTRIBUTE FE2O3 REAL
@ATTRIBUTE FEO REAL
@ATTRIBUTE MNO REAL
@ATTRIBUTE MGO REAL
@ATTRIBUTE CAO REAL
@ATTRIBUTE NA2O REAL
@ATTRIBUTE K2O REAL
@ATTRIBUTE P2O5 REAL
@ATTRIBUTE CO2 REAL
@ATTRIBUTE H20a REAL
@ATTRIBUTE H2Ob REAL
@ATTRIBUTE RKNUM {400,530,570}
@DATA
(..... datos listados.....)

Merece ser discutido como se a interpretado la no aparición de datos en algunos análisis, dado que no en todos los registros aparecía completa. En unos pocos casos, básicamente muestras que se habían utilizado sólo para análisis isotópicos para su datación, no aparecen los óxidos más importantes para su clasificación según los métodos vigentes, como el SiO_2, Al_2O_3, álcalis, CaO , MgO y Fe_2O_3. En estos casos se ha suprimido el registro, no usándose en la construcción del modelo. En el resto de casos, cuando no aparece el valor de TiO_2, MnO, P_2O_5 o fluidos, se ha asignado al óxido ausente el valor = 0.00. Esto se justifica por un lado por el escaso peso clasificatorio de estos óxidos en los esquemas de clasificación tradicional, por su bajo contenido real en la mayoría de las rocas (con frecuencia, por debajo del 1%), por ser su inclusión en la composición parcialmente posterior a la cristalización del magma que las originó (es posiblemente el caso de parte de los fluidos por alteración de la roca tras su aparición en procesos de hidratación y carbonatación), y por que en algunos casos

incluso puede haberse renunciado a analizarse los mismos dado que se presupone valores cercanos a cero o ser de poco interés su estudio.

En la tablas 7.1 puede verse el número de registros finalmente seleccionado correspondientes a cada clase:

RKNUM=400	RKNUM=530	RKNUM=570	TOTAL
203	372	330	905

Tabla 7.1. Número de instancias seleccionadas para cada clase principal del tetraedro de Yoder and Tiller .

La lectura de los registros buscados y de sus atributos, así como las correcciones previamente descritas se ha llevado a cabo mediante la ayuda de la aplicación **pr3.exe,** desarrollada también con Visual C++ 6.0 para DOS. Esta aplicación selecciona los registros de la IGBADAT5.DAT correspondientes a los valores de la variable de la base de datos RKNUM = {400,530,570} y crea un archivo **igbaRes.arff** con los datos de los catorce óxidos mayoritarios del análisis químico y el valor de RKNUM para cada registro seleccionado. También genera el archivo **igbaRes.txt** con registros seleccionables en principio, pero con datos analíticos "anómalos" a priori. Estos registros se han descartado finalmente por corresponder a rocas muy alteradas químicamente o a rocas sin análisis de componentes mayoritarios[19] antes mencionadas.

Listado codigo fuente de **pr3.exe** en **pr3.cpp**

```
// pr3.cpp : Defines the entry point for the console application.
//pre: existe el fichero "c:\\tfc\\igbadat5.dat" con el formato esperado
//     de acuerdo a la descripción dada por la IUGS
// post: se crea el fichero   "c:\\tfc\\igbaRes.arff" con los datos analíticos
//de los registros seleccionados de igbadat5.dat. Tambien se construye la
//cabecera del archivo ARFF
//se crea el fichero "c:\\tfc\\igbaRes.res" en el cual se situan los registros
//      con datos analíticos anómalos. Las muestra con contenidos en SiO2 inferiores al 35%
//se sitúan en el mismo, pero no en el fichero ARFF

#include "stdafx.h"
#include "pr3.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

#define CERO "   "
#define CERONUM "     "
#define BASALTOALCALINO 400
#define TOLEITA 570
#define BASALTOLIVINICO  530

/////////////////////////////////////////////////////////////////////////////
// The one and only application object
```

[19] El resto de casos, aunque en la base de datos se indica que la roca se halla alterada, se ha conservado el registro para disponer de un mayor número de instancias para construir el modelo. Si se dispusiera de más registros se podría "a priori" mejorar el modelo suprimiendo rocas significativamente alteradas.

```
CWinApp theApp;

// LISTA FUNCIONES AUXILIARES
void convertir(char *s,char *vector);
int convertirNum(char *s);

using namespace std;

/*************************************************************/
// FUNCION PRINCIPAL

int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
                                                int nRetCode = 0;
// en el tipo Dato se escribe los datos analíticos de la roca, su RKNUM
// y su identificación en IGBADAT5
struct Dato{
char id[6];
int codRoca;
char SiO2[5];
char TiO2[5];
char Al2O3[5];
char Fe2O3[5];
char FeO[5];
char MnO[5];
char MgO[5];
char CaO[5];
char Na2O[5];
char K2O[5];
char P2O5[5];
char CO2[5];
char H2OA[5];
char H2OB[5];
} ;

Dato dato; //crea una estructura dato
CFile fp1; //puntero a c:\\tfc\\igbadat5.dat
CFile fp2; //puntero a c:\\tfc\\igbaRes.arff
CFile fp3; //puntero a c:\\tfc\\igbaRes.res
char tmpbuf[81];
char aux[5]=CERO; // se usa para conversión a dato numérico
char auxNum[6];
char buffer_cabecera[300]; // string para la cabecera ARFF
int i=0;
int j=0;
CFileException fileException;

// initialize MFC and print and error on failure

            if (!AfxWinInit(::GetModuleHandle(NULL), NULL, ::GetCommandLine(), 0))
{
// TODO: change error code to suit your needs
cerr << _T("Fatal Error: MFC initialization failed") << endl;
nRetCode = 1;
                                                }
else
{
                        // TODO: code your application's behavior here.

             if ( !fp1.Open( "c:\\tfc\\igbadat5.dat", CFile::modeRead, &fileException ) )
```

```
{

   TRACE( "No se puede abrir %s, error = %u\n","c:\\tfc\\igbadat5.dat", fileException.m_cause );
nRetCode=1;
return nRetCode; // error: no existe igbadat5.dat de acuerdo a la pre-condición
}
   if ( !fp2.Open( "c:\\tfc\\igbaRes.arff", CFile::modeCreate+CFile::modeWrite, &fileException ) )
                                                                {

  TRACE( "No se puede abrir %s, error = %u\n","c:\\tfc\\igbaRes.arff", fileException.m_cause );
fp1.Close();
nRetCode=1;
return nRetCode;// error: no se puede crear  igbaRes.arff
}
if ( !fp3.Open( "c:\\tfc\\igbaRes.txt", CFile::modeCreate+CFile::modeWrite, &fileException ) )
                                                                                {
   TRACE( "No se puede abrir %s, error = %u\n","c:\\tfc\\igbaRes.txt", fileException.m_cause );
fp1.Close();
fp2.Close();
nRetCode=1;
return nRetCode;// error: no se puede crear  igbaRes.txt
                                                                                }

//* codigo de lectura
UINT lectura;
  sprintf(buffer_cabecera,"@relation basalto\n@ATTRIBUTE SIO2 REAL\n@ATTRIBUTE TIO2
REAL\n@ATTRIBUTE AL2O3 REAL\n@ATTRIBUTE FE2O3 REAL\n@ATTRIBUTE FEO
REAL\n");
fp2.Write(buffer_cabecera,strlen(buffer_cabecera));
sprintf(buffer_cabecera,"@ATTRIBUTE MNO REAL\n@ATTRIBUTE MGO REAL\n@ATTRIBUTE CAO
REAL\n@ATTRIBUTE NA2O REAL\n@ATTRIBUTE K2O REAL\n@ATTRIBUTE P2O5 REAL\n");
fp2.Write(buffer_cabecera,strlen(buffer_cabecera));
sprintf(buffer_cabecera,"@ATTRIBUTE CO2 REAL\n@ATTRIBUTE H20a REAL\n@ATTRIBUTE H2Ob
REAL\n@ATTRIBUTE RKNUM {400,530,570}\n@DATA\n");
fp2.Write(buffer_cabecera,strlen(buffer_cabecera));
// escribe la cabecera en el puntero de archivo ARFF en fp2

do{
// se leen bloques de 80 caracteres de fp1
lectura=fp1.Read(tmpbuf,sizeof(tmpbuf));
tmpbuf[80]='\0';
i++;
if((lectura==sizeof(tmpbuf))&&(tmpbuf[5]=='B')) // busca línea tipo 'B'
{
j++;
//lee la identificación de la roca y la escribe en la estructura Dato dato
strncpy(dato.id,tmpbuf,5);
dato.id[5]='\0';
// lee RKNUM
strncpy(auxNum,&tmpbuf[71],5);
auxNum[5]='\0';
// convierte el dato RKNUM a tipo entero
dato.codRoca=convertirNum(auxNum);
//comprueba si la roca es un basalto 530,570 o 400
if((dato.codRoca==BASALTOALCALINO)||(dato.codRoca==BASALTOLIVINICO)||(dato.codRoca==TOLEITA))
{
//se escriben los datos analíticos seleccionados en la estructura Dato
//para cada componente químico
            strncpy(aux,&tmpbuf[10],4);
            aux[4]='\0';
```

```
            convertir(aux,dato.SiO2);
            strncpy(aux,&tmpbuf[14],4);
            aux[4]='\0';
            convertir(aux,dato.TiO2);
            strncpy(aux,&tmpbuf[18],4);
            aux[4]='\0';
            convertir(aux,dato.Al2O3);
            strncpy(aux,&tmpbuf[22],4);
            aux[4]='\0';
            convertir(aux,dato.Fe2O3);
            strncpy(aux,&tmpbuf[26],4);
            aux[4]='\0';
            convertir(aux,dato.FeO);
            strncpy(aux,&tmpbuf[30],4);
            aux[4]='\0';
            convertir(aux,dato.MnO);
            strncpy(aux,&tmpbuf[34],4);
            aux[4]='\0';
            convertir(aux,dato.MgO);
            strncpy(aux,&tmpbuf[38],4);
            aux[4]='\0';
            convertir(aux,dato.CaO);
            strncpy(aux,&tmpbuf[42],4);
            aux[4]='\0';
            convertir(aux,dato.Na2O);
            strncpy(aux,&tmpbuf[46],4);
            aux[4]='\0';
            convertir(aux,dato.K2O);
            strncpy(aux,&tmpbuf[50],4);
            aux[4]='\0';
            convertir(aux,dato.P2O5);
            strncpy(aux,&tmpbuf[54],4);
            aux[4]='\0';
            convertir(aux,dato.CO2);
            strncpy(aux,&tmpbuf[58],4);
            aux[4]='\0';
            convertir(aux,dato.H2OA);
            strncpy(aux,&tmpbuf[62],4);
            aux[4]='\0';
            convertir(aux,dato.H2OB);
if(j<90000&&(atof(dato.SiO2)>35.0)){
//si no hay errores analíticos se formatean y escriben los datos
printf("i=%d\n",i); //salida por pantalla, para control visual
            // se escriben los datos formateados en el archivo ARFF
            fp2.Write(dato.SiO2,5);
            fp2.Write(", ",2);
            fp2.Write(dato.TiO2,5);
            fp2.Write(", ",2);
            fp2.Write(dato.Al2O3,5);
            fp2.Write(", ",2);
            fp2.Write(dato.Fe2O3,5);
            fp2.Write(", ",2);
            fp2.Write(dato.FeO,5);
            fp2.Write(", ",2);
            fp2.Write(dato.MnO,5);
            fp2.Write(", ",2);
            fp2.Write(dato.MgO,5);
            fp2.Write(", ",2);
            fp2.Write(dato.CaO,5);
            fp2.Write(", ",2);
```

```
            fp2.Write(dato.Na2O,5);
            fp2.Write(", ",2);
            fp2.Write(dato.K2O,5);
            fp2.Write(", ",2);
            fp2.Write(dato.P2O5,5);
            fp2.Write(", ",2);
            fp2.Write(dato.CO2,5);
            fp2.Write(", ",2);
            fp2.Write(dato.H2OA,5);
            fp2.Write(", ",2);
            fp2.Write(dato.H2OB,5);
            fp2.Write(", ",2);
            sprintf(aux,"%5d",dato.codRoca);
            fp2.Write(aux,5);
            fp2.Write("\n",1);}
else
{
// para control y posterior decisión de inclusión de la muestras
// con datos analíticos presuntamente anómalos
            fp3.Write(dato.SiO2,5);
            fp3.Write(", ",2);
            fp3.Write(dato.TiO2,5);
            fp3.Write(", ",2);
            fp3.Write(dato.Al2O3,5);
            fp3.Write(", ",2);
            fp3.Write(dato.Fe2O3,5);
            fp3.Write(", ",2);
            fp3.Write(dato.FeO,5);
            fp3.Write(", ",2);
            fp3.Write(dato.MnO,5);
            fp3.Write(", ",2);
            fp3.Write(dato.MgO,5);
            fp3.Write(", ",2);
            fp3.Write(dato.CaO,5);
            fp3.Write(", ",2);
            fp3.Write(dato.Na2O,5);
            fp3.Write(", ",2);
            fp3.Write(dato.K2O,5);
            fp3.Write(", ",2);
            fp3.Write(dato.P2O5,5);
            fp3.Write(", ",2);
            fp3.Write(dato.CO2,5);
            fp3.Write(", ",2);
            fp3.Write(dato.H2OA,5);
            fp3.Write(", ",2);
            fp3.Write(dato.H2OB,5);
            fp3.Write(", ",2);
            sprintf(aux,"%5d, linea =%d",dato.codRoca,i);
            fp3.Write(aux,strlen(aux));
            fp3.Write("\n",1);}
}
}
//se efectua el bucle do-while hasta que no se alcanza el final de  fichero
}while(lectura==sizeof(tmpbuf));
fp1.Close();
fp2.Close();
fp3.Close();
}
return nRetCode;
}
```

```
/*****************************************************************/

// función que convierte una cadena de caracteres a tipo numérico punto flotante
// para unificar el formato de la misma a dos decimales e igualar la precisión de los datos analíticos
// pre: *vector es un puntero a una cadena de al menos 5 caracteres
// *s contiene una cadena alfanumérica convertible con un máximo de cinco cifras o caracteres blancos
// no pudiendo existir ningún otro tipo de carácter
//post: se retorna el valor convertido a cadena de caracteres  formateado con dos decimales en el parámetro
// *vector

void convertir(char *s,char *vector)
{
double f;
if (!strcmp(s,CERO))
        {
// cinco caracteres blancos, se interpreta como cero
sprintf(vector,"00.00");
return; //devuelve cero en precisión de dos decimales
        }
 if(s[2]==' ')
        {
//corrige la precisión del dato sin decimales
  f=atof(s);
sprintf(vector,"%0#5.2f",f);
    return;
        }
 if(s[3]==' ')
        {
//corrige la precisión del dato de un sólo decimal
  f=atof(s);
   f=f/10.0;
sprintf(vector,"%0#5.2f",f);
    return;
        }
  f=atof(s);
  f=f/100.0;
sprintf(vector,"%0#5.2f",f);
   return ;
}

/*****************************************************************/
//** convierte una cadena a tipo entero
/** pre: *s contiene una cadena convertible **/
/** post: se devuelve el valor convertido como entero**/
int convertirNum(char *s)
{
                        if (!strcmp(s,CERONUM)) return 0;
                                return atoi(s);

                                }


/*****************************************************************/
```

8 CONSTRUCCIÓN DEL MODELO: PRUEBAS Y ANÁLISIS DE RESULTADOS.

8.1 Arquitectura del modelo.

La arquitectura general de las pruebas para la definición del modelo de red neural más apropiado tendrá las siguientes características:

- Construcción de redes neurales de tres capas**: capa de input, capa oculta y capa de output.**
- La capa de output estará constituida por tres neuronas, correspondiendo cada neurona a una de las tres clases posibles clases de la clasificación de Yoder and Tiller.
- Las neuronas de la capa de input corresponderán a los atributos de entrada seleccionados en las diferentes pruebas.
- Los análisis se efectúan usando las 905 instancias válidas disponibles tras el tratamiento de los datos.
- Se usan siempre inputs normalizado entre –1.0 y +1.0 del valor de los atributos numéricos.
- El tiempo del análisis debe ser computacionalmente razonable.
- El método de evaluación ,por defecto, del modelo será utilizar el 66% de los registros como conjunto de entrenamiento seleccionadas automáticamente por el programa, siendo el resto utilizado como conjunto de evaluación. No se usa un tercer conjunto para test de validación. El resultado de la selección del conjunto de evaluación puede verse en la siguiente tabla:

RKNUM=400	RKNUM=530	RKNUM=570	TOTAL
76	115	117	308

Tabla 8.1. Número de instancias seleccionadas para cada clase principal del tetraedro de Yoder and Tiller en el conjunto de evaluación en todas las pruebas.

En los siguientes apartados se indica un resumen de lo resultados obtenidos en las diferentes pruebas y los parámetros utilizados.

8.2 Modelos básicos: uso de todos los atributos y determinación del número de épocas apropiado en los test.

En este conjunto de cinco pruebas se intenta determinar el número necesario de neuronas en la capa oculta que deben utilizarse para obtener el modelo. Asimismo, se pretende determinar el número suficiente de épocas que deben utilizarse en el cálculo para obtener un modelo ajustado de pesos sin que aparezca sobreentrenamiento. Se usan como entrada todos los atributos disponibles. El resto de parámetros, el momento y la ratio de aprendizaje, se dejan constantes, usándose los valores por defecto propuestos por el menú de WEKA. A continuación se listan los parámetros utilizados en cada prueba y los resultados obtenidos en el output de WEKA. Se ha optado en este punto que no se incluya en la misma los valores obtenidos de los pesos para facilitar la lectura.

PRUEBA 1

ATRIBUTOS	**Metodología test de prueba**	**momento**	**ratio aprendizaje α**	**número de épocas**
todos (15)	**training set 66% test set 33%**	**0.2**	**0.3**	**1000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**8**	**No**	**70.12**	

Output de weka

```
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2          TIO2          AL2O3         FE2O3         FEO           MNO           MGO
              CAO           NA2O          K2O           P2O5          CO2           H20a          H2Ob
              RKNUM

Test mode:    split 66% train, remainder test
ime taken to build model: 7.19 seconds

=== Evaluation on test split ===
=== Summary ===
Correctly Classified Instances         216               70.1299 %
Incorrectly Classified Instances        92               29.8701 %
Kappa statistic                          0.5348
Mean absolute error                      0.2168
Root mean squared error                  0.3828
Relative absolute error                 49.9392 %
Root relative squared error             81.606  %
Total Number of Instances              308
=== Confusion Matrix ===
   a   b   c   <-- classified as
  28  33  15 |   a = 400
  14  81  20 |   b = 530
   3   7 107 |   c = 570
```

PRUEBA 2

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos (15)	**training set 66% test set 33%**	**0.2**	**0.3**	**1000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**73.37**	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 1000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM

Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         226               73.3766 %
Incorrectly Classified Instances        82               26.6234 %
Kappa statistic                          0.5882
Mean absolute error                      0.1851
Root mean squared error                  0.3934
Relative absolute error                 42.6394 %
Root relative squared error             83.8573 %
Total Number of Instances              308

=== Confusion Matrix ===

   a   b   c   <-- classified as
  40  23  13 |   a = 400
  15  84  16 |   b = 530
   0  15 102 |   c = 570
```

PRUEBA 3

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos (15)	**training set 66% test set 33%**	**0.2**	**0.3**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**74.67**	

Output de weka

```
= Run information ===

Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         230               74.6753 %
Incorrectly Classified Instances        78               25.3247 %
Kappa statistic                          0.6084
Mean absolute error                      0.1726
Root mean squared error                  0.3907
Relative absolute error                 39.7396 %
Root relative squared error             83.2886 %
Total Number of Instances              308

=== Confusion Matrix ===

  a   b   c   <-- classified as
 38  28  10 |   a = 400
 13  92  10 |   b = 530
  4  13 100 |   c = 570
```

PRUEBA 4

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos (15)	training set 66% test set 33%	0.2	0.3	3000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	50	No	74.67	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 50
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         230               74.6753 %
Incorrectly Classified Instances        78               25.3247 %
Kappa statistic                          0.6068
Mean absolute error                      0.1702
Root mean squared error                  0.3832
Relative absolute error                 39.2029 %
Root relative squared error             81.6914 %
Total Number of Instances              308

=== Confusion Matrix ===

  a   b   c   <-- classified as
 36  29  11 |   a = 400
  9  93  13 |   b = 530
  4  12 101 |   c = 570
```

PRUEBA 5

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos (15)	training set 66% test set 33%	0.2	0.3	6000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	25	No	74.02	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 6000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         228               74.026  %
Incorrectly Classified Instances        80               25.974  %
Kappa statistic                          0.5988
Mean absolute error                      0.1717
Root mean squared error                  0.39
Relative absolute error                 39.5474 %
Root relative squared error             83.1245 %
Total Number of Instances              308

=== Confusion Matrix ===

   a   b   c   <-- classified as
  39  27  10 |   a = 400
  14  89  12 |   b = 530
   4  13 100 |   c = 570
```

8.3 Afinamiento del modelo. Pruebas con reducción del número de atributos

En este conjunto de tres pruebas se intenta determinar si se puede optimizar el modelo reduciendo el número de atributos del mismo. Se usan tres criterios diferentes, correspondientes a la estrategia de cada prueba:

- Selección de atributos mediante las herramientas proporcionadas por WEKA en la prueba 6
- Uso sólo de los óxidos fundamentales en la composición de los minerales del grupo de los silicatos esperables en la composición de los basaltos, así como en la obtención de sus minerales normativos en la prueba 7. Se excluyen los fluidos.
- Uso de los óxidos utilizados en los diagramas *silice vs álcalis* (Prueba 8)

El resto de parámetros se deja constante y se aprovecha los resultados obtenidos en las pruebas anteriores en la determinación del número de neuronas y de épocas necesarias para construir el modelo,

En la prueba 6 se ha seleccionado los atributos con el auxilio del programa selector incorporado en el paquete WEKA. Un análisis de los datos con el selector , usando el método *Best first*, genera el resultado siguiente, en el cual ser seleccionan sólo siete óxidos como significativos en el modelo.

```
== Attribute Selection on all input data ===

Search Method:
Best first.
Start set: no attributes
Search direction: forward
Stale search after 5 node expansions
Total number of subsets evaluated: 106
Merit of best subset found:    0.306
Attribute Subset Evaluator (supervised, Class (nominal): 15 RKNUM):
CFS Subset Evaluator
Selected attributes: 1,4,7,9,10,11,13 : 7
                     SIO2
                     FE2O3
                     MGO
                     NA2O
                     K2O
                     P2O5
                     H20a
```

PRUEBA 6

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
SIO2, FE2O3, MGO, NA2O, K2O, P2O5, H20a	**training set 66% test set 33%**	**0.2**	**0.3**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**71.75**	

Output de weka

```
=== Run information ===

Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto-weka.filters.AttributeFilter-V-R1,4,7,9-11,13,15
Instances:    905
Attributes:   8
              SIO2
              FE2O3
              MGO
              NA2O
              K2O
              P2O5
              H20a
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         221               71.7532 %
Incorrectly Classified Instances        87               28.2468 %
Kappa statistic                          0.5618
Mean absolute error                      0.1958
Root mean squared error                  0.4227
Relative absolute error                 45.1014 %
Root relative squared error             90.1124 %
Total Number of Instances              308

=== Confusion Matrix ===

  a   b   c   <-- classified as
 38  24  14 |   a = 400
 13  80  22 |   b = 530
  0  14 103 |   c = 570
```

PRUEBA 7

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
SIO2, Al2O3, FE2O3, CaO MGO, NA2O, K2O,	**training set 66% test set 33%**	**0.2**	**0.3**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**69.15**	

Output de weka

```
=== Run information ===

Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto-weka.filters.AttributeFilter-V-R1,3-4,7-10,15
Instances:    905
Attributes:   8
              SIO2
              AL2O3
              FE2O3
              MGO
              CAO
              NA2O
              K2O
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         213               69.1558 %
Incorrectly Classified Instances        95               30.8442 %
Kappa statistic                          0.5266
Mean absolute error                      0.2066
Root mean squared error                  0.4131
Relative absolute error                 47.5868 %
Root relative squared error             88.0479 %
Total Number of Instances              308

=== Confusion Matrix ===

  a  b  c   <-- classified as
 40 28  8 |  a = 400
 21 74 20 |  b = 530
  6 12 99 |  c = 570
```

PRUEBA 8

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
SIO2, NA2O, K2O,	training set 66% test set 33%	0.2	0.3	3000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**65.90**	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto-weka.filters.AttributeFilter-V-R1,9-10,15
Instances:    905
Attributes:   4
              SIO2
              NA2O
              K2O
              RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         203               65.9091 %
Incorrectly Classified Instances       105               34.0909 %
Kappa statistic                          0.4721
Mean absolute error                      0.2686
Root mean squared error                  0.4097
Relative absolute error                 61.8596 %
Root relative squared error             87.336  %
Total Number of Instances              308

=== Confusion Matrix ===

  a   b   c   <-- classified as
 34  33   9 |   a = 400
 20  65  30 |   b = 530
  0  13 104 |   c = 570
```

8.4 Pruebas efectuadas con diferentes valores del coeficiente de aprendizaje y momento. Prueba variando el tamaño del conjunto de entrenamiento.

En este conjunto de pruebas se intenta determinar la influencia de los valores dados a los parámetros ratio de aprendizaje y momento en los resultados obtenidos y comprobar si modificando los mismos se puede obtener mejora en el modelo. (Pruebas 9 al 12)

PRUEBA 9

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	training set 66% test set 33%	0.2	0.1	3000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	25	No	76.62	

Output de weka

```
=== Run information ===

Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.1 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2          TIO2          AL2O3         FE2O3         FEO           MNO
              MGO           CAO           NA2O          K2O           P2O5          CO2           H20a
              H2Ob          RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         236               76.6234 %
Incorrectly Classified Instances        72               23.3766 %
Kappa statistic                          0.6374
Mean absolute error                      0.1703
Root mean squared error                  0.3786
Relative absolute error                 39.21   %
Root relative squared error             80.7026 %
Total Number of Instances              308

=== Confusion Matrix ===

  a   b   c   <-- classified as
 40  29   7 |   a = 400
  7  96  12 |   b = 530
  3  14 100 |   c = 570
```

PRUEBA 10

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	training set 66% test set 33%	0.05	0.3	3000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	25	No	75.00	

Output de weka

```
== Run information ===

Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.3 -M 0.05 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2            TIO2            AL2O3           FE2O3           FEO             MNO
              MGO             CAO             NA2O            K2O             P2O5            CO2
              H20a            H2Ob            RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         231               75      %
Incorrectly Classified Instances        77               25      %
Kappa statistic                          0.6134
Mean absolute error                      0.1663
Root mean squared error                  0.3773
Relative absolute error                 38.3025 %
Root relative squared error             80.4317 %
Total Number of Instances              308

=== Confusion Matrix ===

  a  b  c   <-- classified as
 40 24 12 |  a = 400
 12 92 11 |  b = 530
  3 15 99 |  c = 570
```

PRUEBA 11

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	**training set 66% test set 33%**	**0.00**	**0.05**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**74.35**	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.05 -M 0.0 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2            TIO2            AL2O3           FE2O3           FEO             MNO
              MGO             CAO             NA2O            K2O             P2O5            CO2
              H20a            H2Ob            RKNUM
Test mode:    split 66% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         229               74.3506 %
Incorrectly Classified Instances        79               25.6494 %
Kappa statistic                          0.6021
Mean absolute error                      0.1945
Root mean squared error                  0.3694
Relative absolute error                 44.7927 %
Root relative squared error             78.7358 %
Total Number of Instances              308

=== Confusion Matrix ===

   a   b   c   <-- classified as
  36  29  11 |   a = 400
  11  93  11 |   b = 530
   3  14 100 |   c = 570
```

PRUEBA 12

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	training set 66% test set 33%	0.1	0.05	3000
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	25	No	75.32	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.05 -M 0.1 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM
Test mode:    split 66% train, remainder test
Time taken to build model: 382.67 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         232               75.3247 %
Incorrectly Classified Instances        76               24.6753 %
Kappa statistic                          0.6186
Mean absolute error                      0.1856
Root mean squared error                  0.3658
Relative absolute error                 42.7449 %
Root relative squared error             77.9807 %
Total Number of Instances              308

=== Confusion Matrix ===

   a   b   c   <-- classified as
  42  23  11 |   a = 400
  12  90  13 |   b = 530
   2  15 100 |   c = 570
```

Finalmente, en las pruebas 13 y 14 se intenta mejorar el modelo usando más instancias de registros para construir el conjunto de entrenamiento del modelo (el 80% de las mismas) y menos para construir el conjunto de evaluación.

PRUEBA 13

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	**training set 80% test set 20%**	**0.1**	**0.05**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**76.24**	

Output de weka

```
Scheme:       weka.classifiers.neural.NeuralNetwork -L 0.05 -M 0.1 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2          TIO2          AL2O3         FE2O3         FEO           MNO
              MGO           CAO           NA2O          K2O           P2O5          CO2
              H20a          H2Ob          RKNUM
Test mode:    split 80% train, remainder test

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         138               76.2431 %
Incorrectly Classified Instances        43               23.7569 %
Kappa statistic                          0.6324
Mean absolute error                      0.1723
Root mean squared error                  0.3677
Relative absolute error                 39.5502 %
Root relative squared error             78.252  %
Total Number of Instances              181

=== Confusion Matrix ===

  a  b  c   <-- classified as
 23 16  5 |  a = 400
  6 55  2 |  b = 530
  0 14 60 |  c = 570
```

PRUEBA 14

ATRIBUTOS	Metodología test de prueba	momento	ratio aprendizaje α	número de épocas
todos	**training set 80% test set 20%**	**0.2**	**0.1**	**3000**
Decaimiento	**Neuronas en la capa oculta**	**test de validación**	**Resultado: % de instancias clasificada correctamente del test set**	
No	**25**	**No**	**75.13**	

Output de weka

```
cheme:      weka.classifiers.neural.NeuralNetwork -L 0.1 -M 0.2 -N 3000 -V 0 -S 0 -E 20 -H 25
Relation:     basalto
Instances:    905
Attributes:   15
              SIO2
              TIO2
              AL2O3
              FE2O3
              FEO
              MNO
              MGO
              CAO
              NA2O
              K2O
              P2O5
              CO2
              H20a
              H2Ob
              RKNUM
Test mode:    split 80% train, remainder test
ime taken to build model: 605.33 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         136               75.1381 %
Incorrectly Classified Instances        45               24.8619 %
Kappa statistic                          0.6125
Mean absolute error                      0.1805
Root mean squared error                  0.3937
Relative absolute error                 41.4179 %
Root relative squared error             83.7924 %
Total Number of Instances              181
=== Confusion Matrix ===

  a  b  c   <-- classified as
 19 21  4 |  a = 400
  3 56  4 |  b = 530
  0 13 61 |  c = 570
```

9 Análisis de resultados de las pruebas y conclusiones

1) En las pruebas iniciales se obtienen los mejores resultados utilizando un número de 25 neuronas en la capa oculta. No se obtiene ninguna mejora significativa al utilizar más neuronas. Se acepta este número como suficiente.

2) Aumentado hasta 3000 épocas el número de ciclos de entrenamiento de los pesos se mejoran los resultados sin que haya una excesiva penalización en cuanto a tiempo de ejecución del programa. Usando para efectuar la prueba un Pentium IV de con frecuencia de reloj de 1.2 GHZ en entorno W98 se ha obtenido el resultado en menos de diez minutos. Se aprecia también que aumentando el número de épocas a partir de este punto no mejora el resultado significativamente, apareciendo claramente sobreentrenamiento del modelo cuando se usan 6000 épocas.

3) No se observa mejoría de resultados disminuyendo el número de atributos en ningún caso.

4) Existe una leve mejoría en la convergencia del modelo al utilizar valores menores de ratio de aprendizaje y momento.

5) No se consigue una mejora al aumentar el tamaño del conjunto de entrenamiento hasta el 80% de los registros disponibles.

6) En general, una vez determinado el valor del número de épocas y del número de neuronas de la capa oculta, las mejoras introducidas variando el resto de parámetros son poco significativas.

7) El modelo que presenta mejores resultados de convergencia, con un 76.62% de instancias correctamente clasificadas es el resultado de la prueba 9. Los pesos de dicho modelo se listan en el Anexo II del documento adjunto a la memoria

8) Se obtendría probablemente resultados mejores si se partiera de la aceptación universal del dominio de las clases del modelo clasificatorio por parte de los diversos autores que aportaron sus análisis a la base de datos. En los valores de la matriz de confusión se puede observar como en prácticamente todas las pruebas obtienen mejores resultados para las clases **toleíta (570)** y **basalto olivínico (530),** mientras que la clase **basalto alcalino (400)**, término que en la literatura geológica puede haberse usado en más de un contexto diferente, presenta peores resultados. Por ejemplo, en la prueba 9, las instancias de ésta última en el conjunto de evaluación apenas se clasifican correctamente en el 52% de los casos, pero el 83% del basalto olivínico y el 85% de las toleitas se han clasificado correctamente.[20]

[20] Véase la matriz de confusión del output de WEKA de la prueba.

9) Los resultados no mejoran tampoco al aumentar el tamaño del conjunto de entrenamiento a costa del conjunto de evaluación, como se observa en las pruebas en que se usa el 80% de las instancias para construir el mismo.

10) Se puede concluir que los métodos de clasificación usando redes neuronales pueden ser de aplicación interesante en problemas del tipo “obtener clasificación de objetos y situaciones geológicas a partir de parámetros descriptivos” con modelo físico del proceso geológico que los origina insuficientemente conocido en disciplinas de la geología clásica, complementando o substituyendo los métodos tradicionales de estudio de la misma. En todo caso se requiere evitar posibles solapamientos acusados del dominio de las clases, por lo cual hay que estar muy atento a la calidad de los datos utilizados como entrada en el modelo y a los criterios de como se han obtenido los mismos. Un buen conocimiento de ambas cosas puede permitir el uso de mejores estrategias en la obtención de modelos.

11) WEKA se ha demostrado como una buena y flexible herramienta en la construcción de estos modelos. Sin embargo presenta alguna carencia, como la imposibilidad de modificar la forma de la función de activación, aunque a priori esto pueda ser poco importante [Masters,1993] . En este estudio sólo se ha podido utilizar funciones de activación lineales.

Bibliografía

Brändle, J.L. and Nagy,G (1995). *The state of 5th version of IGBA: igneous petrological data base***.** Computers and Geociences**.** Vol 21. Nº 3. pp. 425-432

Ceballos,Javier (2003). *Programación orientada a objetos con C++***.** Ed. Ra-Ma**.** Madrid (1a Edición)

Chayes, Felix (1986). *IGBADAT:A World Data Base for Igneous Petrology***.** Episodes, V. 8, nº 4. pp 245-251

D'Andrea, Edgar . (2000). *Visual C++ 6.0. Guía Completa***.** Inforbooks S.L Editores. Barcelona

Gurney, Kevin (1997). *An Introduction to Neural Networks.* Ed. CRC Pres. Londres.(Repr. 2003)

Hall, Anthony. (1998). *Igneous Petrology.2nd Edition.* Ed. Longman. Essex

Hurlbut,JR, and Cornelis,K .(1985). *Manual de mineralogía de Dana.* Editorial Reverté SA . Barcelona

Kamtath, Chandrika (2001). *On Mining Scientific Datasets*. En Data Mining for Scientific and Engineering Applications. pp 1 – 21. Grossman et al. Eds. Kluwer Academic Publishers.

Lees, Brian (1996). *Neural Network applications in the Geosciences: an introduction.* Computer and Geosciences, V.22, No 9, pp 955-957

Masters, Timothy (1993). *Practical Neural Network Recipes in C++*. Academic Press. Morgan Kaufman Publishers.

Witten, I. and Frank, E. (2000) . *Data mining: practical machine learning tools and techniques with Java implementations*. Academic Press. Morgan Kaufman Publishers

Anexo I. Códigos numéricos del sistema de clasificación IUGS para las rocas ígneas.

10 NOT NAMED IN SOURCE
20 NOT NAMED IN IGBA

30 ABSAROKITE	430 - ANDESITE	840 CHARNOCKITE
40 ADAMELLITE	440 - ANKARAMITIC	850 CHROMITITE
50 AGGLOMERATE	450 - CALCALKALINE	860 CIMINITE
60 AGPAITE	460 - DOLERITIC	870 CINERITE
70 ARERITE	470 - ESSEXITE	880 COMEMDITE
80 ALASKITE	480 - FERRO-	890 - TRACHYTIC
90 ALBANITE	490 - HIGH-ALUMINA	900 CRAIGNURITE
100 ALBITITE	500 - HYPERSTHENE	910 CRINANITE
105 ALBITOPHYRE	510 - LATITE	920 CUMULATE
110 ALBORANITE	520 - MUGEARITE	930 DACITE
120 ALEUTITE	530 - OLIVINE	940 - ANDESITE
130 ALGARVITE	540 - PICRITE	950 - CALCALKALINE
140 ALLIVALITE	550 - QUARTZ	960 - THOLEIITIC
150 ALNOITE	560 - SPILITIC	970 DELLENITE
	565 - SUBALKALI	
160 CARBONATITE	570 - THOLEIITIC	980 DIABASE
170 AMPHIBOLITE	580 - THOL.-PICRITE	990 - ALKALI
180 ANALCIMITE	590 - TRACHYANDESITE	1000 - SPILITIC
190 ANDESITE	600 - TRANSITIONAL	1010 - THOLEIITIC
200 - BASALTIC	610 - 2 PYROXENE	1020 DIALLAGITE
210 - HIGH-ALUMINA	620 BASANITE	1030 DIORITE
220 - LATITE	630 - PHONOLITIC	1040 - MICRO-
230 - THOLEIITIC	640 BASANITOID	1050 - QUARTZ
240 - 2 PYROXENE	650 BEFORSITE	1060 DOLERITE
250 ANKARAMITE	660 BEKINKINITE	1070 - ALKALI
260 ANKARATRITE	670 BENMOREITE	1074 - META-
270 ANORTHOSITE	680 - PHONOLITIC	1075 - PEGMATITIC
280 APHANITE	690 BERGALITE	1080 - QUARTZ
290 APLITE	700 BERONDRITE	1090 DOMITE
300 APLODIORITE	710 BIOTITITE	1100 DOREITE
310 APLOGRANITE	720 BLAIRMORITE	1110 DUNITE
320 APORHYOLITE	730 BOMB	1120 ECOLOGITE
330 APPINITE	740 BOROLANITE	1130 EKERITE
340 ASH	750 BOSTONITE	1140 ELVAN
350 ATLANTITE	760 - QUARTZ	1150 ENSTATITITE
360 AUGITITE	770 BRONZITITE	1160 EPIDIORITE
370 BANAKITE	780 BUCHITE	1170 ESSEXITE
380 BANDAITE	790 BUCHONITE	1180 - QUARTZ
390 BASALT	800 CAMPANITE	1190 ETINDITE
400 - ALKALI	810 CAMPTONITE	1200 ETNAITE
410 - ALKALI OLIVINE	820 CARBONATITE	1210 EUCRITE
420 - ALKALI PICRITE	830 CECILITE	1220 FARSUNDITE

Table A1. Rock Names, cont.

1230 FASINITE	1730 JUMILLITE	2250 MISSOURITE
1240 FELSITE	1740 JUVITE	2260 MONCHIQUITE
1250 FENITE	1750 KAJANITE	2270 MONZODIORITE
1260 FLOW	1760 KAKORTOKITE	2280 - QUARTZ
1270 FORTUNITE	1770 KATUNGITE	2290 MONZOGABBRO
1280 FOURCHITE	1780 KAUAIITE	2300 - QUARTZ
1290 FOYAITE	1790 KENTALLENITE	2310 MONZONITE
1300 GABBRO	1800 KENYITE	2320 - MICRO-
1310 - ALKALI	1810 KERATOPHYRE	2330 - QUARTZ
1320 - ESSEXITE	1820 - QUARTZ	2340 MUGEARITE
1330 - QUARTZ	1830 KERSANTITE	2345 - BASIC
1340 - THERALITE	1840 KIMBERLITE	2350 - SODA
1350 GABBRODIORITE	1850 KIVITE	2360 MURAMBITE
1360 GABBRONORITE	1860 KOMATIITE	2370 MURITE
1370 GAUTEITE	1870 - BASALTIC	2380 NAUJAITE
1380 GIBELITE	1880 - PERIDOTITIC	2390 NEPHELINITE
1390 GLASS	1890 KOTUITE	2400 NEVADITE
1400 GLENMUIRITE	1900 KULAITE	2410 NGURUMANITE
1410 GLIMMERITE	1910 LABRADORITE	2420 NILIGONGITE
1420 GRANITE	1920 LAMPROITE	2430 NORDMARKITE
1430 - ALKALI	1930 LAMPROPHYRE	2440 - MICRO-

1440 - MICRO
1450 - PERALKALINE
1460 - RAPAKIVI
1470 - SODA
1480 - 2 MICA
1485 - GNEISSIC
1490 GRANODIORITE
1500 GRANOGABBRO
1510 GRANOPHYRE
1520 GREISEN
1530 GRORUDITE
1540 GUARDIAITE
1550 HAKUTOITE
1560 HARRISITE
1570 HARZBURGITE
1580 HAUYNITE
1590 HAUNOPHYRE
1600 HAWAIITE
1610 HIGHWOODITE
1620 HORNBLENDITE
1625 HYALOCLASTITE
1630 HYALOTRACHYTE
1640 HYPERITE
1650 HYPERSTHENITE
1660 ICELANDITE
1670 - BASALTIC
1680 IGNIMBRITE
1690 IJOLITE
1700 INNINMORITE
1710 ITALITE
1720 JACUPIRANGITE
1940 LARDALITE
1950 LARVIKITE
1960 LATIANDESITE
1970 LATITE
1980 - QUARTZ
1990 LAVA
2000 LEDMORITE
2010 LEIDLEITE
2020 LEUCITITE
2030 LEUCITOPHYRE
2040 LHERZOLITE
2050 LIMBURGITE
2060 LINOSAITE
2070 LIPARITE
2080 LUGARITE
2090 LUJAVRITE
2100 LUSCLADITE
2110 LUSITANITE
2120 MADUPITE
2130 MAFRAITE
2140 MAGNETITITE
2150 MALIGNITE
2160 MANDSCHURITE
2170 MANGERITE
2180 MARSCOITE
2190 MELILITITE
2200 MELTEIGITE
2210 MIASKITE
2220 MICKENITE
2230 MIMOSITE
2240 MINETTE
2450 - QUARTZ
2460 NORITE
2470 - MICRO-
2480 - QUARTZ
2490 NOSELITITE
2500 OBSIDIAN
2510 - PERALKALINE
2520 OCEANITE
2530 ODINITE
2540 OKAITE
2550 OLIVINITE
2560 ONGONITE
2570 OPHIOLITE
2580 ORDANCHITE
2590 ORENDITE
2600 ORTHOSITE
2610 ORVIETITE
2620 OTTAJANITE
2630 OUACHITITE
2640 PAISANITE
2650 PALAGONITE
2660 PANTELLERITE
2670 PEGMATITE
2680 - MICRO-
2690 PELE'S HAIR
2700 PEPERINO
2710 PERIDOTITE
2720 PERKNITE
2730 PERLITE
2740 PERTHOSITE
2750 PHANERITE

Table A1: Rock Names, cont.

2760 PHONOLITE
2770 - ALKALI
2780 - BASANITIC
2790 - LATITE
2800 - TEPHRITIC
2810 PICOTITITE
2820 PICRITE
2830 PITCHSTONE
2840 PLAGIOGRANITE
2850 PLAGIOLIPARITE
2860 PLAGIOTRACHYTE
2870 PORPHYRY
2880 - FELDSPAR
2890 - QUARTZ
2900 - RHOMB-
2910 PSEUDOTACHYLITE
2920 PULASKITE
2930 PUMICE
2940 PYROXENITE
2950 - CLINO-
2960 - ORTHO-
2970 RAPAKIVI
2980 RAUHAUGITE
2990 RHYOBASALT
3000 RHYODACITE
3010 RHYOLITE
3020 - ALKALI
3030 - CALCALKALINE
3040 - PERALKALINE
3050 - SODA
3060 - THOLEIITIC
3070 - TRACHYTIC
3080 RINGITE
3090 ROCKALLITE
3100 RODINGITE
3110 RONGSTOCKITE
3120 ROUGEMONTITE
3130 RUSHAYITE
3140 SAKALAVITE
3150 SANCYITE
3160 SANIDINITE
3170 - SODA
3180 SANTORINITE
3210 SCORIA
3220 SELBERGITE
3230 SERPENTINITE
3240 SHACKANITE
3250 SHIHLUNITE
3260 SHONKINITE
3270 SHOSHONITE
3280 SIDEROMELANE
3290 SILEXITE
3300 SOLVSBERGITE
3310 SOVITE
3320 SPESSARTITE
3330 SPILITE
3340 SUSSEXITE
3350 SYENITE
3360 - ALKALI
3370 - MICRO-
3380 - NEPHELINE
3390 - PERALKALINE
3400 - QUARTZ
3410 - RAPAKIVI
3420 SYENODIORITE
3430 SYENOGABBRO
3440 TACHYLITE
3450 TAHITITE
3460 TANNBUSCHITE
3470 TAUTIRITE
3480 TEPHRA
3490 TEPHRITE
3500 - ANDESITE
3510 - BASALTIC
3515 - PHONOLITIC
3520 TEPHRITOID
3530 TESCHENITE
3540 - PICRITE
3550 THERALITE
3560 - ESSEXITE
3570 THOLEIITE
3580 - HIGH ALUMINA
3590 - LOW ALUMINA
3600 - OLIVINE
3610 TINGUAITE
3620 TONALITE
3650 TRACHYANDESITE
3660 TRACHYBASALT
3670 TRACHYBASANITE
3680 TRACHYDACITE
3690 TRACHYDOLERITE
3700 TRACHYLIPARITE
3710 TRACHYPHONOLITE
3720 TRACHYTE
3730 - ALKALI
3740 - MUGEARITE
3750 - PANTELLERITIC
3760 - PERALKALINE
3780 - QUARTZ
3790 - RHYOLITIC
3800 - SODA
3810 - TEPHRITIC
3820 TRACHYTEANDESITE
3830 TRACHYVICOITE
3840 TRAP
3850 TRISTANITE
3860 TROCTOLITE
3870 TRONDHJEMITE
3880 TUFF
3890 TURLITE
3900 UGANDITE
3910 ULTRAMAFITE
3920 UMPTEKITE
3930 UNCOMPAHGRITE
3940 URTITE
3950 VARIOLITE
3960 VENANZITE
3970 VERITE
3980 VESUVITE
3990 VICOITE
4000 VITROPHYRE
4010 VOGESITE
4020 - SODA
4030 VULSINITE
4040 WEBSTERITE
4050 WEHRLITE
4060 WELDED TUFF
4070 WOODENITE
4080 WYOMINGITE

```
3190 SANUKITE
3200 SAXONITE
3630 TORDRILLITE
3640 TOSCANITE
4090 YAMASKITE
4100 ZWITTER
```