8.1 – Results summarized for every added feature

As a result of experiments, we have obtained the goal of classification of spam, with 95% accuracy, on this basis of testing, may conduct more experiments in the future for further improvement in the features selection. Whether using SVM (Support Vector Machines) or MLP (Multi-Layer Perceptron).

As can be seen from the graph the learning curve reaches their maximum capacity ranking with only 17 of the 28 features used.

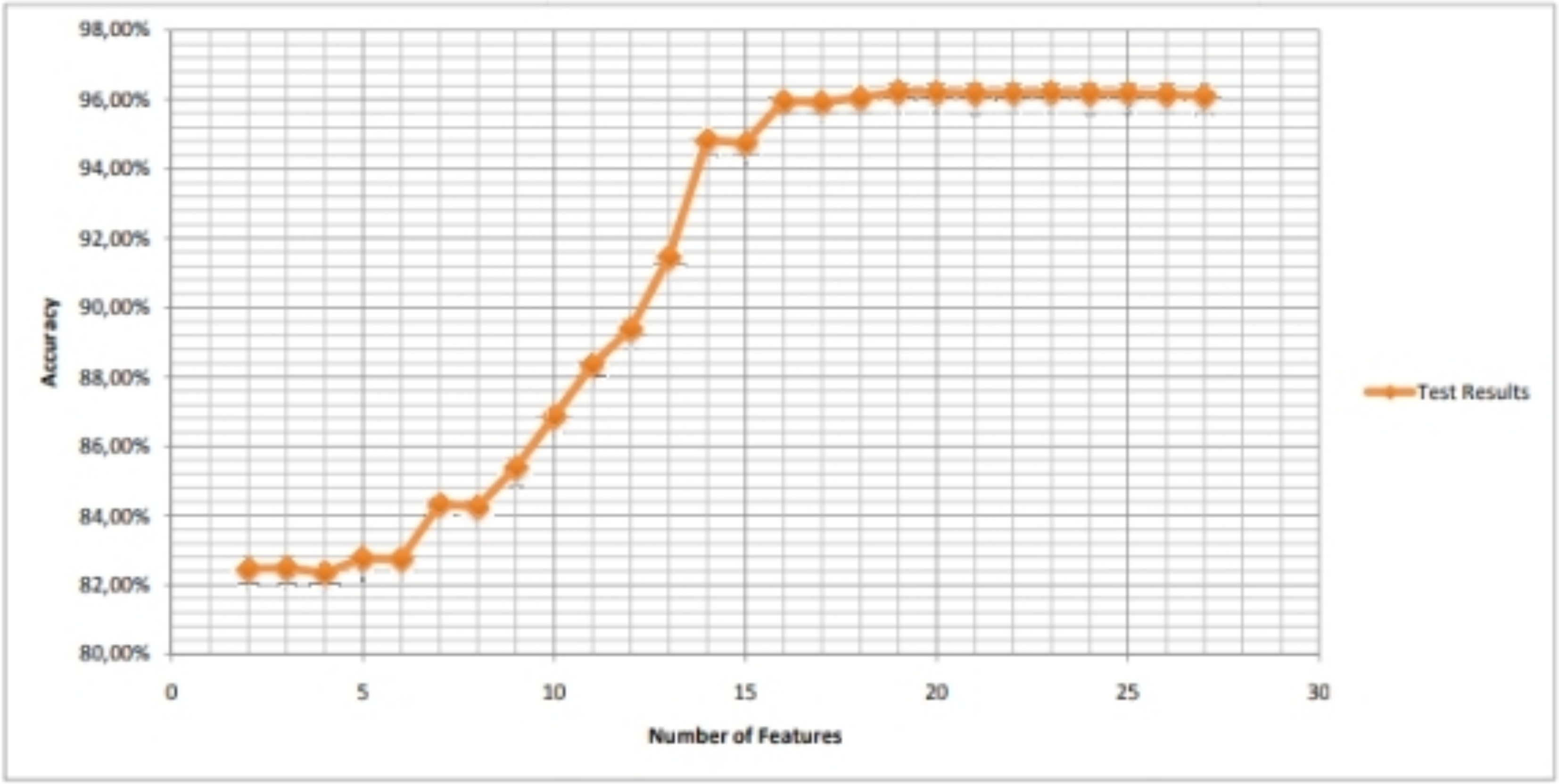
Recall that in the first definition of features, 31 features were studied but only 28 were programmed to perform the feature selection experiments.

With this we can show that for a proper features search, able to make a generalization is necessary work on features selection to allow as much accuracy as possible with the least computational cost.

Make a selection feature generalization Improve the performance by a reduction of the dimensionality and by eliminating irrelevant variables.
Given a set of features, select a subset that performs best under the evaluation criteria, in our case using single-layer artificial neural networks as Classifiers.

Our investigation shows that the forward feature selection algorithms improves the efficiency with not too much computational cost, while does not corrupt the correctness of the selected feature data set so that the prediction accuracy using the algorithm have made the experimental results remains satisfactory.

Features Classification Results



9 – Discussions, Conclusions & Future work

For "the corpus" has been necessary to install a complex system within an ISP in Spain, to collect a large number of "real" emails that recreate in the laboratory of what is happening in the e-mails in Spain and to detect and visualize both the emails themselves, and the structure of spams and hams, to learn and study the characteristics of the email , head and body.

The corpus used in the experiments is 100% real different from the corpus used by other papers studied and annexed in the references of this work, these corpus, have a single sender or recipient. The universe used in the laboratory experiments is 100% real and represents "real" emails where senders and recipients are quite different.

An anti-spam filtering system was proposed which uses the artificial neural network trained by the backpropagation algorithm. The results clearly show that the Subject and Body fields can contain enough information for spam classification in order to obtain near 95% prediction accuracy.

Although the NN technique is accurate and useful, its spam precision performance is not high enough for it to be used without supervision. For this technique to be more useful, the feature set would require additional members or modifications. It should be noted, however, that the NN required fewer features to achieve results similar to the Naive Bayesian approaches, indicating that descriptive qualities of words and messages, similar to those used by human readers, can be used effectively to distinguish spam by a classifier.

For future work, we suggest that a combination of keywords and descriptive characteristics may provide more accurate classification, as well as the combine of spam classificators techniques. Also we pretend to implement this work in the real world using this features and ANN trained to see how is the potential for spam classification and how to improve the accuracy near 99% and the CPU performance.

A neural network classifier using these descriptive features, however, may not degrade over time as rapidly as classifiers that rely upon a relatively static vocabulary from spammers. Strategies that apply a combination of techniques, such as a NN with whitelist, would likely yield better results.

We have interest in continuing professional development of this thesis work, incorporating new features that provide artificial neural networks, as well as a self-learning system that allows work in the real world on the basis of new features and classification techniques spam The ultimate goal would be to obtain a 99,9% of maximum accuracy.

10 – Bibliography References

1- Abduelbaset M. Goweder, Tarik Rashed , Ali S. Elbekaie, and Husien A. Alhammi, "An anti-spam system using artificial neural networks and genetic algorithms", Proc. Int. Arab Conf. on Information Technology, 2008.

2- Akaike:1973 - H. Akaike. Information theory as an extension of the maximum likelihood principle. In B.N. Petrov and F. Csaki, editors, *Second International Symposium on Information Theory*, 267-281, Akademiai Kiado, Budapest, 1973.

3- Alexandru Catalin Cosoi, "A False Positive Safe Neural Network; The Followers of the Anatrim Waves", Proc. Spam Conference 2008.

4- Almuallim et al, 91 - H.Almuallim, and T.G.Dietterich. Learning with many irrelevant features. In Porc. AAAI-91, pp 547-552. MIT Press, 1991.

5- Androutsopoulos, I; Koutsias, J; Chandrinos, K. V. and Spyropoulos, C. D. 2000. An experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (Athens, Greece, 2000), pp. 160-167.

6- Bishop, Christopher M. (1996-01-18). *Neural Networks for Pattern Recognition*. Oxford University Press. p. 504. ISBN 0198538642 ISBN 978-0198538646. http://research.microsoft.com/~cmbishop/books.htm.

7- Bishop, Christopher M. (2006-08-17). *Pattern Recognition and Machine Learning*. Springer. p. 738. ISBN 978-0387310732. http://research.microsoft.com/~cmbishop/PRML.htm.

8- Burton, B. 2002. SpamProbe – Bayesian Spam Filtering Tweaks. http://spamprobe.sourceforge.net/paper.html; last accessed November 17, 2003.

9- C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller (1996). "**Context-specific independence in Bayesian networks**." *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI '96)* (pp. 115-123).

10- Caruana et al, 94] R. Caruana, and D. Freitag. Greedy attribute selection. In Proc. ML-94, Morgan Kaufmann, 1994.

11- Cranor, L. F. and LaMacchia, B. A. 1998. Spam! Communications of the ACM, 41(8): pp. 74-83.

12- Declude, IP Lookup Against a List of All Known DNS-based Spam Databases. http://www.declude.com/junkmail/support/ip4r.htm; last accessed January 27, 2004.

13- Graham, P. 2002. A Plan for Spam. http://www.paulgraham.com/spam.html; last accessed November 17, 2003.

14- Graham, P. 2003. Better Bayesian Filtering. In Proceedings of the 2003 Spam Conference (Cambridge, Massachusetts,2003). See http://spamconference.org/proceedings2003.html.

15- Hauser, S. 2002. Statistical Spam Filter Works for Me. http://www.sofbot.com/article/Statistical_spam_filter.html; last accessed November 17, 2003.

16- Hauser. S. 2003. Statistical Spam Filter Review. http://www.sofbot.com/article/Spam_review.html; last accessed November 17, 2003.

17- Haykin, 1994 - Haykin, S. (1994). Neural networks: a comprehensive foundation. Macmillan, New York.

18- James Clark, Irena Koprinska, and Josiah Poon, "A Neural Network Based Approach to Automated E-mail Classification", Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence, IEEE Computer Society.
19- John(1994) G. H. John.
When the best move isn't optimal: Q-learning with exploration.
In Proceedings of the Twelfth National Conference on Artificial Intelligence, page 1464, Seattle, WA, 1994.
20- Jordan, M.I. (1995), "Why the logistic function? A tutorial discussion on probabilities and neural networks", MIT Computational Cognitive Science Report 9503, http://www.cs.berkeley.edu/~jordan/papers/uai.ps.Z
21- Kira et al, 92 K.Kira, and L.A.Rendell, The feature selection problem: traditional methodsand a new algorithms. In Proc. AAAI-92, pp 129-134. MIT Press. 1992
22- Langley et al, 94 - P.Langley, and S.Sage, Induction of selective Bayesian classifiers. In Proc.UAI-94, pp 399-406. Seattle, WA. Morgan Kaufmann, 1994.
23- McCullagh, P. and Nelder, J.A. (1989) Generalized Linear Models, 2nd ed., London: Chapman & Hall.
24- Miller, 1990 - Miller, K. D. (1990). Correlation-based mechanisms of neural development. In Gluck, M. A. and Rumelhart, D. E., editors, Neuroscience and Connectionist Theory, pages 267--353. Erlbaum, Hillsdale NJ.
25- Minsky and Papert, 1969 - Minsky, M. and Papert, S. (1969). Perceptrons. MIT Press, Cambridge.
26- Moore et al, 94 - A.W.Moore, and M.S.Lee, Efficient algorithms for minimizing cross-validation error. In Proc. ML-94, Morgan Kaufmann, 1994.
27- Rosenblatt, 1962 - Rosenblatt, F. (1962). Principles of Neurodynamics. Spartan, New York.
28- Sahami, M.; Dumais, S.; Heckerman, D. and Horvitz, E. 1998. A Bayesian Approach to Filtering Junk E-mail. In Learning for Text Categorization—Papers from the AAAI Workshop (Madison, Wisconsin, 1998), AAAI Technical Report WS-98-05, pp. 55-62.
29- Sebastiani, F. 2002. Machine Learning in Automatic Text Categorization. ACM Computing Surveys (CSUR), 34(1): pp. 1-47.
30- Shopos Inc. 2010 Spam World Report
31- Skalak, 94- D.B.Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms. In Proc. ML-94, Morgan Kaufmann, 1994.
32- Spertus, E. 1997. Smokey: Automatic Recognition of Hostile Messages. In Proceedings of the 14th National Conference on AI and the 9th Conference on Innovative Applications of AI (Providence, Rhode Island, 1997), pp. 1058-1065.
33- Weiss, A. 2003. Ending Spam's Free Ride. netWorker, 7(2): pp. 18-24.
34- W. Fuertes, M. Almache and J. Ruiz, "Clasificador de E-mails Anti-Spam utilizando un Perceptrón Multicapa", Revista de Ciencia y Tecnología de la Escuela Politécnica del Ejército, Volumen 2. Sangolquí, Ecuador, Mayo 2009.

11 - Software References

1- www.sinespam.com (professional project)
2- The "bpmaster" software was created from the PDP software.
The original PDP software can be found here:

http://www.stanford.edu/group/pdplab/resources.html#originalpdp.
The original PDP handbook can be found here:
http://www.stanford.edu/group/pdplab/originalpdphandbook/.
In the Appendix B there is a list of the original commands with their descriptions. Note, however, that it does not include some important commands, such as "set/ envrm", "set/ ensayo", "ensayo" or "backsel".
The current PDP software can be found here:
http://www.stanford.edu/group/pdplab/resources.html.

Annex I – Programs to obtain "the corpus"

Feature selection programs description

Clamav.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CLAMAV","/home/www/dataminig/clamav.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {
                  echo "$file\n";

                  $resultado=exec(CLAMAV.DIRECTORIO.$file);

                  echo $resultado ;

            $sql="insert into data (filename,att_virus)
values('$file','$resultado')";
            $resultado=mysql_query($sql, $Bd );

            }//Endif

}//EndWhile
closedir($fdir);



?>
```

Spamassasin.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CRM","/home/www/dataminig/spamassassin.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);
```

```
while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {
                  echo "$file\n";

                  $resultado=exec(CRM.DIRECTORIO.$file);

                  echo "$resultado\n";

            $sql="update data set att_spamassasin='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

            }//Endif

}//EndWhile
closedir($fdir);



?>
```

Contaracentos.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");

define("CADENAS","áéíóúÁÉÍÓÚñÑ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {
                  echo "Procesando $file\n";

                  $resultado=CuentaCaracteres(DIRECTORIO.$file);

                  echo "RETORNO $resultado\n";

            $sql="update data set att_spanish='$resultado' where
filename='$file'";
            $resultado=mysql_query($sql, $Bd );

            }//Endif
}//EndWhile
closedir($fdir);


function CuentaCaracteres($file)
{

//Leemos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Procesamos en busca de string en CONSTATE CADENAS
$strings=CADENAS;

//Numero de caracteres ha buscar
$reg=strlen($strings);

for ($i=0;$i<$reg;$i++)
{

      //Buscamos caracter
      $registros=explode($strings[$i],$txt);

      $TOTAL=$TOTAL+count($registros);

}//EndFor


//Devolvemos numero registros
return $TOTAL-$reg;

}//EndFunction

?>
```

Contarcaracteresraros.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");

define("CADENAS","¡!·#$€%~&/()=¿?*^{}[]ª\@><¿'´+-_¨,");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {
                    echo "Procesando $file\n";

                    $resultado=CuentaCaracteres(DIRECTORIO.$file);

                    echo "RETORNO $resultado\n";

            $sql="update data set att_char_extend='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

            }//Endif


}//EndWhile
closedir($fdir);


function CuentaCaracteres($file)
{

//Leemos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Procesamos en busca de string en CONSTATE CADENAS
$strings=CADENAS;

//Numero de caracteres ha buscar
$reg=strlen($strings);

for ($i=0;$i<$reg;$i++)
{

      //Buscamos caracter
      $registros=explode($strings[$i],$txt);

      $TOTAL=$TOTAL+count($registros);

}//EndFor


//Devolvemos numero registros
return $TOTAL-$reg;

}//EndFunction

?>
```

Contarstringnospanish.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/contar_string_nospanish.sh ");


//String ha buscar
define("CADENA","kqxzywv");

//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=ProcesaMensaje(DIRECTORIO.$file);

            echo "RESULTADO = $resultado \n";

            $sql="update data set att_char_nospanish='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);

//Normalizamos fichero
function ProcesaMensaje($file)
{
echo "-----\n";
echo "Procesando $file\n";

$tmp="/tmp/nospanish/";
exec("rm -rf $tmp");
mkdir($tmp);
copy($file,$tmp."msg.txt");

//Normalizamos
exec(CMD." $tmp msg.txt");

//Procesamos SOLO ficheros txt
$fdir=opendir($tmp);
while ($file = readdir($fdir))
{
      if(eregi("textfile",$file))
      {
            echo "subfile $file \n";
            $nospanish=$nospanish+CuentaNospanish($tmp.$file);
      }

}//EndWhile

return $nospanish;
```

```
define("CMD","/home/www/dataminig/contar_string_nospanish.sh ");


//String ha buscar
define("CADENA","kqxzywv");

//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=ProcesaMensaje(DIRECTORIO.$file);

            echo "RESULTADO = $resultado \n";

            $sql="update data set att_char_nospanish='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);

//Normalizamos fichero
function ProcesaMensaje($file)
{
echo "-----\n";
echo "Procesando $file\n";

$tmp="/tmp/nospanish/";
exec("rm -rf $tmp");
mkdir($tmp);
copy($file,$tmp."msg.txt");

//Normalizamos
exec(CMD." $tmp msg.txt");

//Procesamos SOLO ficheros txt
$fdir=opendir($tmp);
while ($file = readdir($fdir))
{
      if(eregi("textfile",$file))
      {
            echo "subfile $file \n";
            $nospanish=$nospanish+CuentaNospanish($tmp.$file);
      }

}//EndWhile

return $nospanish;
}//EndFunction


//Contamos palabras con caracteres nospanish
function CuentaNospanish($file)
{
$cadena=CADENA;

//Leemos datos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Descomponemos en palabras
$words=explode(" ",$txt);

$reg=strlen($cadena);

//Procesamos cada palabra si hay algun letra
//kqxzywv
while (list($key, $value) = each($words))
{
      $KK=0;            //Reseteamos

      //quitamos . ya que evita filtrado
      $chkword=trim(str_replace(".","",$value));

      for ($i=0;$i<$reg;$i++)
      {
            if(eregi($cadena[$i],$value))
            {
                  $KK++;
            }

      }//EndFor

//RESET
if(eregi("http",$value))
      $kk=0;
if(eregi("@",$value))
      $kk=0;


      if($KK>1)
      {
            echo "ENCONTRADO =========== $value \n";
            $TOTAL++;
      }

}//EndWhile

return $TOTAL;

}//EndFunction
```

```
?>
```

Contarstringsnowords.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/contar_string_nospanish.sh ");

//Longitud String ha buscar
define("LSTRING","10");

//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=ProcesaMensaje(DIRECTORIO.$file);

            echo "RESULTADO = $resultado \n";

            $sql="update data set att_long_words='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);

//Normalizamos fichero
function ProcesaMensaje($file)
{
echo "-----\n";
echo "Procesando $file\n";

$tmp="/tmp/nospanish/";
exec("rm -rf $tmp");
mkdir($tmp);
copy($file,$tmp."msg.txt");

//Normalizamos
exec(CMD." $tmp msg.txt");

//Procesamos SOLO ficheros txt
$fdir=opendir($tmp);
while ($file = readdir($fdir))
{
      if(eregi("textfile",$file))
```

```
	{
		echo "subfile $file \n";
		$longword=$longword+CuentaLogWord($tmp.$file);
	}
}//EndWhile

return $longword;

}//EndFunction


//Contamos palabras con LOG WORD
function CuentaLogWord($file)
{

//Leemos datos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Descomponemos en palabras
$words=explode(" ",$txt);

$reg=strlen($cadena);

//Procesamos cada palabra mirando su longitud

while (list($key, $value) = each($words))
{
//	echo strlen($value)."  =  ".$value."\n";

	//RESET
	if(eregi("http",$value))
		break;
	if(eregi("@",$value))
		break;

	if((strlen($value)>LSTRING))
		$KK++;

}//EndWhile

return $KK;

}//EndFunction


?>
```

Contarstringallupercase.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/contar_string_nospanish.sh ");
```

```
//Longitud String ha buscar
define("LSTRING","10");

//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=ProcesaMensaje(DIRECTORIO.$file);

            echo "RESULTADO = $resultado \n";

            $sql="update data set
att_char_alluppercase='$resultado' where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);

//Normalizamos fichero
function ProcesaMensaje($file)
{
echo "-----\n";
echo "Procesando $file\n";

$tmp="/tmp/nospanish/";
exec("rm -rf $tmp");
mkdir($tmp);
copy($file,$tmp."msg.txt");

//Normalizamos
exec(CMD." $tmp msg.txt");

//Procesamos SOLO ficheros txt
$fdir=opendir($tmp);
while ($file = readdir($fdir))
{
      if(eregi("textfile",$file))
      {
            echo "subfile $file \n";
            $longword=$longword+CuentaAllUpper($tmp.$file);
      }
}//EndWhile

return $longword;

}//EndFunction
```

```
//Contamos palabras con todas mayusculas
function CuentaAllUpper($file)
{

//Leemos datos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Descomponemos en palabras
$words=explode(" ",$txt);

$reg=strlen($cadena);

//Procesamos cada palabra comparando la misma en UpperCase
while (list($key, $value) = each($words))
{

      //Quitamos espacios y palabras menores de 2 letras
      $value=trim($value);
      if(strlen($value)<2)
            break;

      $testword=strtoupper($value);

      if($value==$testword)
      {
            echo $value."\n";
            $KK++;
      }//EndIf

}//EndWhile

return $KK;

}//EndFunction


?>
```

Contastringhref.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");

define("STRING","href");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {
                  echo "Procesando $file\n";

                  $resultado=CuentaString(DIRECTORIO.$file);

                  echo "RETORNO $resultado\n";

            $sql="update data set att_body_href='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif


}//EndWhile
closedir($fdir);


function CuentaString($file)
{

//Leemos
$fp=fopen($file,"r");
$txt=fread($fp,filesize($file));
fclose($fp);

//Procesamos en busca de string en CONSTATE CADENAS

      //Buscamos caracter
      $registros=explode(STRING,$txt);

      $TOTAL=$TOTAL+count($registros);



//Devolvemos numero registros
return $TOTAL-1;

}//EndFunction

?>
```

dcc.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/dcc.sh ");


unlink("/tmp/alldcc.log");
```

```
//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
    if ($file !='.')
        if ($file !='..')
        {
            echo "$file\n";

            $resultado=exec(CMD.DIRECTORIO.$file);

            echo "$resultado\n";

        $sql="update data set att_dcc='$resultado' where
filename='$file'";
        $resultado=mysql_query($sql, $Bd );

        }//Endif

}//EndWhile
closedir($fdir);



?>
```

Razor.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/razor.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
    if ($file !='.')
        if ($file !='..')
        {
            echo "$file\n";

            $resultado=exec(CMD.DIRECTORIO.$file);

            echo "$resultado\n";

        $sql="update data set att_razor='$resultado' where
filename='$file'";
        $resultado=mysql_query($sql, $Bd );

        }//Endif
}//EndWhile
closedir($fdir);


?>
```

Pyzor.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/pyzor.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
        if ($file !='.')
                if ($file !='..')
                {
                        echo "$file\n";

                        $resultado=exec(CMD.DIRECTORIO.$file);

                        echo "$resultado\n";

                $sql="update data set att_pyzor='$resultado' where
filename='$file'";
                $resultado=mysql_query($sql, $Bd );

                }//Endif

}//EndWhile
closedir($fdir);


?>
```

Bodyhref.php

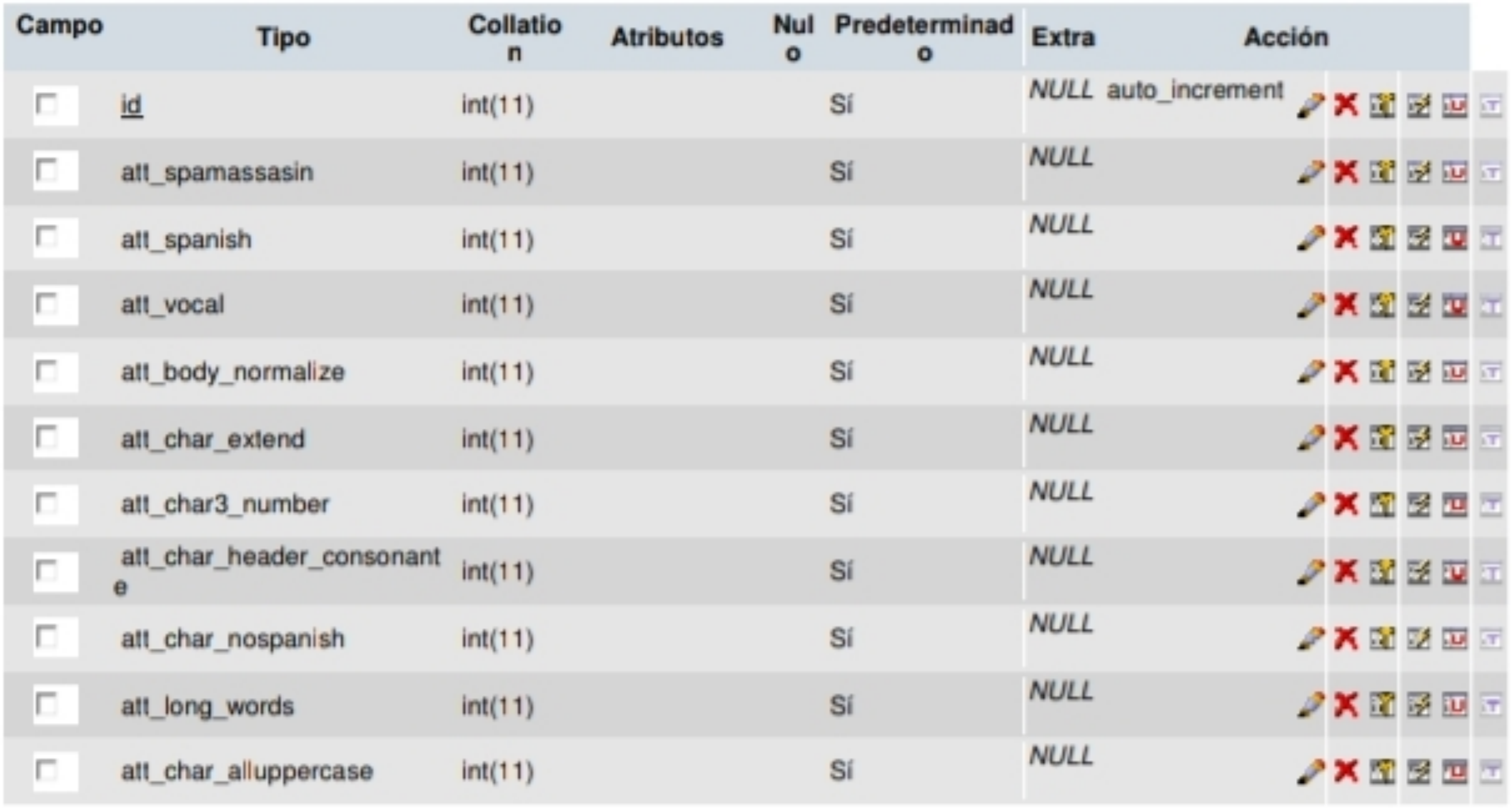
```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/body_href_extend.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
```

```
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=exec(CMD.DIRECTORIO.$file);

            echo "RESULTADO $file = $resultado \n";

            $sql="update data set
att_body_href_extend_char='$resultado' where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);


?>
```

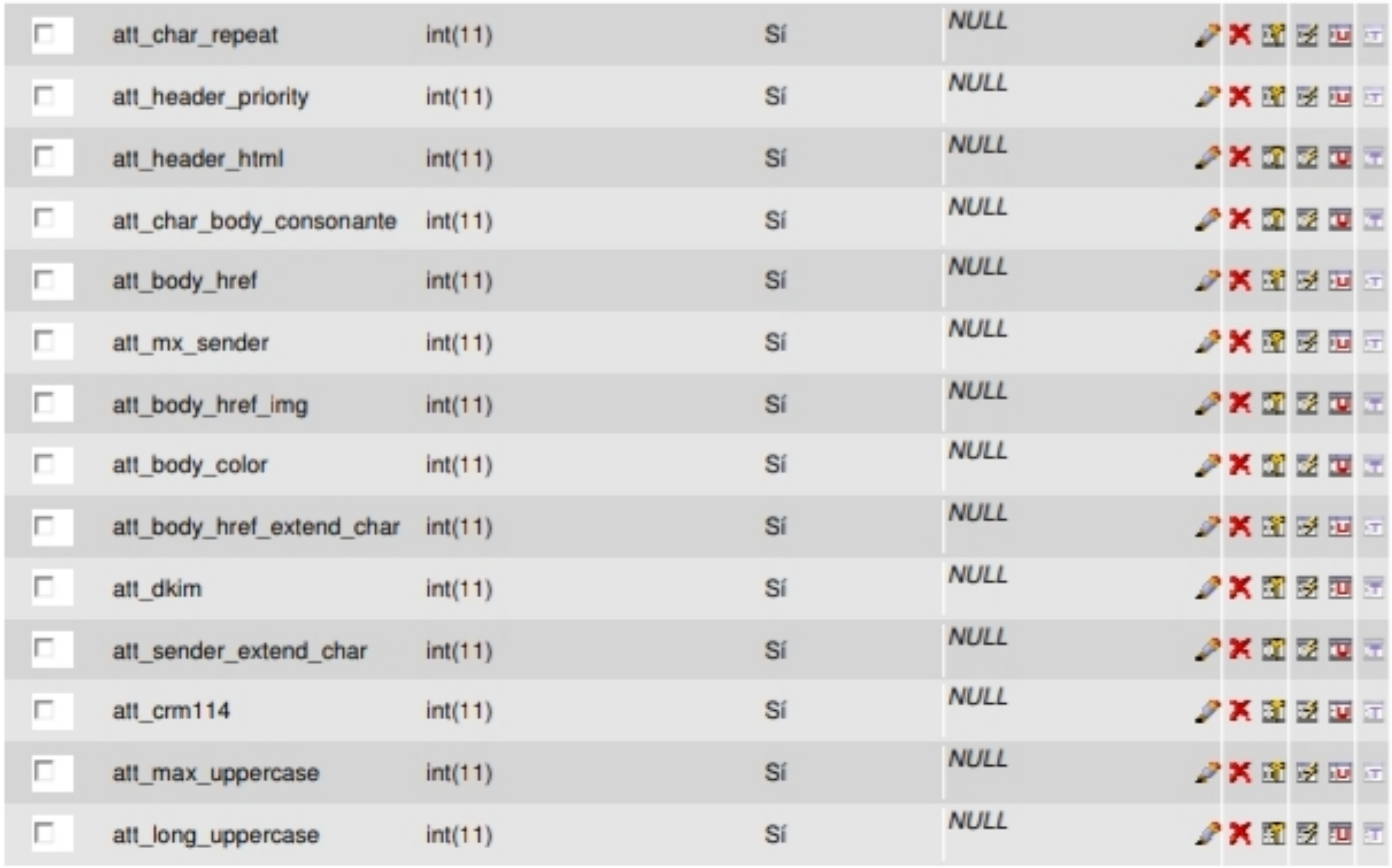
Bodyimghref.php

```
<?
include("dbconfig.php");

define("DIRECTORIO","/home/backup/TESTALL/");
define("CMD","/home/www/dataminig/body_img_href.sh ");


//Leemos ficheros
$fdir=opendir(DIRECTORIO);

while ($file = readdir($fdir))
{
      if ($file !='.')
            if ($file !='..')
            {

            $resultado=exec(CMD.DIRECTORIO.$file);

            echo "RESULTADO $file = $resultado \n";

            $sql="update data set att_body_href_img='$resultado'
where filename='$file'";
            $resultado=mysql_query($sql, $Bd );

//exit;
            }//Endif

}//EndWhile
closedir($fdir);


?>
```

Annex III – DataBase Definition using MySQL



Campo	Tipo	Collation	Atributos	Nulo	Predeterminado	Extra	Acción
☐	id	int(11)			Sí	*NULL* auto_increment	
☐	att_spamassasin	int(11)			Sí	*NULL*	
☐	att_spanish	int(11)			Sí	*NULL*	
☐	att_vocal	int(11)			Sí	*NULL*	
☐	att_body_normalize	int(11)			Sí	*NULL*	
☐	att_char_extend	int(11)			Sí	*NULL*	
☐	att_char3_number	int(11)			Sí	*NULL*	
☐	att_char_header_consonante	int(11)			Sí	*NULL*	
☐	att_char_nospanish	int(11)			Sí	*NULL*	
☐	att_long_words	int(11)			Sí	*NULL*	
☐	att_char_alluppercase	int(11)			Sí	*NULL*	



☐	att_char_repeat	int(11)	Sí	*NULL*
☐	att_header_priority	int(11)	Sí	*NULL*
☐	att_header_html	int(11)	Sí	*NULL*
☐	att_char_body_consonante	int(11)	Sí	*NULL*
☐	att_body_href	int(11)	Sí	*NULL*
☐	att_mx_sender	int(11)	Sí	*NULL*
☐	att_body_href_img	int(11)	Sí	*NULL*
☐	att_body_color	int(11)	Sí	*NULL*
☐	att_body_href_extend_char	int(11)	Sí	*NULL*
☐	att_dkim	int(11)	Sí	*NULL*
☐	att_sender_extend_char	int(11)	Sí	*NULL*
☐	att_crm114	int(11)	Sí	*NULL*
☐	att_max_uppercase	int(11)	Sí	*NULL*
☐	att_long_uppercase	int(11)	Sí	*NULL*

Annex IV – Feature Classification maximum accuracy

FINAL MAXIMUM CLASSIFICATION USING 13 FEATURES

```
**********************************************************

******** RESULTADOS MEDIOS DE LOS ULTIMOS 100 ENSAYOS ********

**********************************************************

--------- RESULTADOS MEDIOS MAXIMOS POR ERROR MEDIO --------

MAXIMOS EN EL CONJUNTO DE TRAIN

 Numero de resultados obtenidos: 100

 epoch: 400.000000 +/-0.00

 TRAIN       tss: 41.04 +/-2.58 (0.03)  aciertos: 1273.34/1320.00 (96.47% +/-0.36%)

   Porcentajes medios de aciertos de cada clase:  97.87%  95.06%

 VALIDATION  tss: 15.18 +/-1.97 (0.03)  aciertos: 422.61/440.00 (96.05% +/-0.64%)

   Porcentajes medios de aciertos de cada clase:  97.64%  94.47%

 TEST        tss: 15.18 +/-1.97 (0.03)  aciertos: 422.65/440.00 (96.06% +/-0.64%)

   Porcentajes medios de aciertos de cada clase:  97.63%  94.49%


MAXIMOS EN EL CONJUNTO DE VALIDATION

 Numero de resultados obtenidos: 100

 epoch: 388.630000 +/-42.54


 TRAIN       tss: 41.26 +/-2.54 (0.03)  aciertos: 1273.01/1320.00 (96.44% +/-0.35%)

   Porcentajes medios de aciertos de cada clase:  97.86%  95.02%

 VALIDATION  tss: 15.16 +/-1.96 (0.03)  aciertos: 422.68/440.00 (96.06% +/-0.62%)

   Porcentajes medios de aciertos de cada clase:  97.64%  94.50%

 TEST        tss: 15.23 +/-1.95 (0.03)  aciertos: 422.66/440.00 (96.06% +/-0.65%)

   Porcentajes medios de aciertos de cada clase:  97.65%  94.48%
```